ANALYSIS AND IMPLEMENTATION OF WIGNER DISTRIBUTION BASED SPECTRAL ESTIMATION TECHNIQUES FOR TIME-VARYING SIGNALS

By

JOELLEN WILBUR

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN
PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

1987

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Fred Taylor for his continuing help and support over the years. He was always available to answer questions and provide assistance and maintained a sincere interest in my professional development. For that I am grateful.

I would like to thank Dr. D. G. Childers for serving on all my committees and for allowing me access to his computer which enabled me to experiment on the Wigner distribution.

I would also like to thank Dr. A. A. Arroyo, Dr. J. C. Principe, and Dr. G. Logothetis for taking the time to serve on my supervisory committee.

I would like to thank Dr. Dave Chester for the numerous hours spent working with me at the start of this study and the continued support and assistance throughout my studies.

I would like to thank Dr. F. Taylor, Dr. D. Childers, Dr. L. Couch and Dr. D. Chester for their assistance in helping me to obtain a university faculty position.

I would like to thank my parents for funding my entire undergraduate program, enabling me to concentrate on my studies and prepare for graduate school.

Finally,I would like to thank my husband who has always supported my studies and has been my primary motivation throughout both my undergraduate and graduate studies.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Abstract of Dissertation Presented to the Graduate School of the University of Florida in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

ANALYSIS AND IMPLEMENTATION OF WIGNER DISTRIBUTION BASED SPECTRAL ESTIMATION TECHNIQUES FOR TIME-VARYING SIGNALS

By

JOELLEN WILBUR

August 1987

Chairman: Dr. Fred J. Taylor

Major Department: Electrical Engineering

A spectral estimator, called the Wigner distribution (WD), is investigated. The WD in the context of the time-frequency representations of signals and its relation to alternative t-f representations is examined. The relationship between the WD as a simultaneous t-f signal representation and the one-dimensional time and frequency domains of deterministic signals forms an important part of this discussion. The effects of moving averages performed

on the WD are considered in terms of windowing and filtering. Window and filter selections for maximum signal concentration are determined.

The WD is applicable to analysis on continuous signals. The discrete-WD (DWD) is the discrete analog. The properties of the DWD and its relation to a digital signal processing (DSP) setting are investigated. A detailed comparison between the DWD and the conventional FFT based moving window method is examined both experimentally and theoretically on signal types whose dispersion index is large. The smoothed-pseudo-DWD is related to filtering and modulation operations on the signal then experimentally shown to improve estimation for the two primary conditions under which the DWD produces poor results: multi-component signals and low signal-to-noise ratios. These results are applied to estimation of a multi-path radio example in the presence of severe receiver noise. Emphasis is placed on the DWD as a DSP tool and the theoretical development of the DWD is heavily supplemented with physical examples and applications directly related to signal processing and analysis. Where appropriate, physical discussions replace or supplement formal mathematical descriptions.

Wigner processor configurations tailored to signal type and system application are provided. General Wigner processing using standard FFTs are addressed and a means of

computing two N point DWD time slices via one N/2 point FFT is applied to the case of real signals. A high-speed DWD (and spDWD) processor for complex data is developed. The processor is based on a single-modulus quadratic residue numbering system where computational reduction can be achieved via an algebraic mapping. Comparison to a conventional processor equivalent is made. A band-selectable DWD for high-frequency spectral analysis is developed. An optical guided-wave WD processor configuration is also presented.

INTRODUCTION

The concept of a time-varying signal, or one with a changing spectrum, has been given considerable attention. From Fourier transform theory it is known that frequency is defined over an infinite time interval and time over an infinite frequency interval. Therefore, the concept of a joint time-frequency representation of a signal is not feasible. Still, a representation of signal concentration over time and frequency serves a useful purpose, an example being the case of phase modulated signals. As a result, there have been attempts to define such a representation based on the hypothesis of minimum spread. In 1952 Page [1] mathematically defined the instantaneous power spectrum of a signal. His representation relied completely on past histories of the signal. In 1963 [2] Levin modified the work of Page to produce an instantaneous spectrum that was equivalent to the double Fourier transform of the complex conjugate of the ambiguity function. In 1967 Rihaczek [3] combined the one-dimensional energy densities in the time

and frequency domains to produce a combined time-frequency representation which he defined as the complex energy density of the signal. In 1966, Cohen[4] developed a generalized class of two-dimensional representations relating electron position and momentum as applied to statistical mechanics then defined a set of correspondence rules associating the marginal distributions of position and momentum. The extension of the Cohen class to signal analysis, as pointed out by DeBruijn [5], is clear. The relation between position and momentum of an electron, and time and frequency of a signal are equivalent (i.e. they both satisfy the Fourier transform relationship).

In this regard, any member of the Cohen class may be viewed as an intermediary between the time and frequency domains. Ideally, all properties defining this class are preserved by the chosen representation. Naturally, such a case is not possible as the properties are self contradictory (a result of Heisenberg's uncertainty in Fourier analysis as established by Schrodinger). Still, most standard time-frequency representations are members of the generalized class given by Cohen, where the Cohen class and its ideal set of properties establishes a basis on which to relate time-frequency representations of a signal [5-11].

In general, most joint time-frequency signal representations are obtained via short-time Fourier

transform (STFT) techniques. Fourier transforms are taken with respect to the center position of a sliding window, ensemble averaged, and used to characterize the spectrum. Such STFT methods postulate spectral stationarity over the observation interval. This defines a compromise between frequency resolution and adequate preservation of the nonstationarities in the signal dispersion characteristics. Despite optimum windowing attempts, STFT techniques generally produce a representation that is overly smoothed in both time and frequency. Accordingly, few of the Cohen signal properties are preserved by STFT derived representations. Instead, most properties are averaged over the observation interval.

One signal representation that does not necessitate such a compromise is the Wigner distribution (WD). The WD satisfies all of the properties in the Cohen class except that of positivity. Viewed as a link between the time and frequency domains, common signal time domain properties (instantaneous power, signal energy, instantaneous frequency, and average frequency) and common frequency domain properties (energy spectral density, signal energy, group delay, and average time delay) are intrinsic to the WD. The WD is in fact the only member of the class of time-frequency representations to precisely maintain these properties. In fact, all other members of the Cohen class can be represented by weighted averages of the WD.

The Wigner distribution (WD) was initially developed by Wigner and used as a tool for formulating correction terms to the Boltzman transport equations relevant to low temperatures [12]. Despite the presence of negative values, Wigner established the WD as an indicator of the joint probability of position and momentum. DeVille first applied the WD to signal description [13]. At a later time, DeBruijn set forth a firm mathematical foundation for the WD[5,6]. Recently, Claassen and Mecklenbrauker established the WD and the discrete WD (DWD) to be a viable signal processing tool [7,14,15]. Their development paralleled that of DeBruijn, where a mathematical examination of the WD and its properties formed the primary emphasis of their study. Since that time several mathematical descriptions have been used to investigate the WD and its associated properties [16-23], and specific applications of the WD have been explored.

Chester examined the DWD as a speech analysis tool by comparing DWD representations of selected word utterances with their associated spectrogram [24]. Janse and Kaizer applied the WD to loudspeakers design [8] and Preis examined the role of the WD and associated moments that prove important to audio signal processing [25]. Martin and Flandrin used a smoothed-psuedo-WD, where a moving average was performed on the WD kernel, to study biological rhythms of caverniculous animals [26]. The spWD was also

applied to bat captures by Fandrin and Martin. The multidimensional WD and spWD have been considered in the contexts of optics and image processing [27-29]. The WD has been suggested for signal synthesis and filter design applications as well [30,31].

The increased interest in the WD as a signal processing/analysis tool strongly indicates the need for a thorough physical investigation of the WD and its relation to signal types in a SP and DSP setting. Without this unifying framework as a basis, applications and studies of the WD, although interesting, are often inconclusive.

One of the primary objectives of this dissertation will be to develop such a framework for the WD. An in depth study of the WD, and variations thereof, with respect to its role in the analysis and description of signals will form the first phase of this dissertation. Both DeBruijn [5,6] and later Claassen and Mecklenbrauker [7,14,15] have provided a formal mathematical development of the WD, and others have added to these initial studies [7-11,16-23]. Chapters 2 and 3 of this dissertation will attempt to unify and add to the most important of these contributions into one comprehensive study as applies to SP and DSP.

Chapter 2 examines where the Wigner distribution derives from in regard to time-frequency representations of signals and its relation to alternative t-f representations. The relationship between the WD as a

simultaneous t-f signal representation and the one-dimensional time and frequency domains of deterministic signals forms an important part of this discussion. The effects of moving averages performed on the WD are considered in terms of windowing and filtering to produce representations that maintain weighted averages of the time and frequency domain properties. Window and filter selection for maximum signal concetration are determined.

Chapter 3 will apply the results of Chapter 2 to a DSP setting. The computational efficiency of FFT data processing makes short-time Fourier transform (STFT) techniques, despite their disadvantages, attractive for real-time implementation. The DWD provides an alternative FFT based estimation technique. The DWD is the DSP analogue to the WD and the correspondence of the continuous case properties, etc. in a DSP setting will be investigated. A detailed comparison between the DWD and the conventional FFT based moving window method, known as the STP, will be given. The DWD will be performed on a large class of signal types whose dispersion index is large and compared with results obtained via the STP.

Generally in DSP applications, a windowed version of the WD, termed the psuedo-DWD, is computed resulting in a frequency smoothed DWD. Flandrin[9] offered a more generalized expression termed the smoothed-psuedo-DWD (spDWD) where a weighted average of the DWD kernel is used

to reduce frequency smoothing. The purpose of the spDWD was to effect cross-term reduction in multi-component signals. In Chapter 3 the spDWD is interpreted in terms of filtering and modulation operations. In this form the wealth of knowledge on filtering and modulation may be applied to the spDWD to improve estimation for the two primary conditions under which the DWD produces poor results: multi-component signals and low signal-to-noise ratios.

Two primary advantages result from the spDWD: first, relaxation of the time/frequency smoothing trade-off associated with the STP; and, second the ability to effect smoothing similar to that performed on the STP (to improve performance) without the computational burden of computing multiple FFTs at each time slice. These claims are experimentally tested and verified. As a final example, the spDWD is applied to estimation of a multipath radio example in the presence of severe receiver noise.

Chapter 4 addresses means of implementing the WD and associated variations of the WD provided in Chapters 2 and 3. Processor configurations tailored to signal type and system application are provided. General Wigner processing using standard FFTs are addressed and a means of computing two N point DWD time slices via one N/2 point FFT is applied to the case of real signals. A high-speed DWD (and spDWD) processor for complex data is developed. The

processor is based on a single-modulus quadratic residue system numbering where computational reduction can be achieved via an algebraic mapping. Comparison to a conventional processor equivalent is made. A band-select DWD for high-frequency analysis is developed. An optical guided-wave WD processor configuration is presented.

The overall objective of this dissertation is to provide, in a self-contained manner, sufficient preparation to effectively use the WD for signal processing and analysis applications. Emphasis is placed on the DWD (and spDWD) as a DSP tool and the theoretical development of the DWD is heavily supplemented with physical examples and applications directly related to signal processing and analysis. Where appropriate, physical discussions replace or supplement formal mathematical descriptions.

CHAPTER 2
WIGNER DISTRIBUTION--A SPECTRAL ESTIMATOR

2.1 Time-frequency Representations

Most joint time-frequency representations used in the analysis of signals, whose spectra vary with time, are members of the generalized class of two dimensional representations established by Cohen [4] and defined as

$$C_f(t,\omega)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(u+\tau/2)f^*(u-\tau/2)\phi(\xi,\tau)e^{j(\xi t-\tau\omega-\xi u)}\,du\,d\tau\,d\xi \qquad (2\text{-}1)$$

where $f(t)$ and $f^*(t)$ are the respective time signal and its complex conjugate and $\phi(t,\omega)$ is an arbitrary kernel that corresponds to the particular signal "distribution" to be considered. The term "distribution" is set off in quotes to emphasize its loose usage. Assuming $C_f(t,\omega)$ is formally a distribution of the signal, then $C_f(t,\omega)$ should be positive over the entire t-ω plane and integration should produce the correct marginal distributions. With respect

to signal analysis these are

$$\int_{-\infty}^{\infty} C_f(t,\omega;\phi)d\omega = |f(t)|^2 \qquad (2\text{-}2)$$

and

$$\int_{-\infty}^{\infty} C_f(t,\omega;\phi)dt = |F(\omega)|^2. \qquad (2\text{-}3)$$

Equation (2-2) states that the integral of $C_f(t,\omega)$ over all frequencies at any fixed time yields the instantaineous power at that time; (2-3) states that the integral over all time at any frequency yields the energy density at that frequency. Equations (2-2) and (2-3) indicate that integration of $C_f(t,\omega)$ over the whole time-frequency plane gives the total energy in the signal. However, if $C_f(t,\omega)$ is observed in terms of signal energy, sharp discrimination in both time and frequency conflicts with the relation $\Delta f \sim 1/\Delta t$ where Δt denotes the time during which the signal experiences appreciable change and Δf the corresponding band of frequencies.

As an example, let $C_f(t,\omega)=S_f(t,\omega)$, where $S_f(t,\omega)$ denotes the well-known short-time spectral density (STSD).

The STSD is defined as the square modulus of the short-time Fourier transform (STFT) over all possible window positions. The STFT expresses as

$$F_t(\omega)=\int_{-\infty}^{\infty} e^{-j\omega\tau} f(\tau)h(\tau-t)d\tau, \qquad (2-4)$$

where h(t) denotes the moving window and is centered about t. The power spectrum then expresses as

$$S(t,\omega)= |F_t(\omega)|^2. \qquad (2-5)$$

Integrating $S(t,\omega)$ over time at a fixed frequency produces an averaging of the energy density over the window as follows

$$\int_{-\infty}^{\infty} S(t,\omega)dt=(1/2\pi)\int_{-\infty}^{\infty} |F(\zeta)|^2 |H(\omega-\zeta)|^2 d\zeta. \qquad (2-6)$$

If, instead, $C_f(t,\omega)$ is treated as an intermediary between the time and frequency domains, signal properties which characterize the respective one-dimensional time and

frequency domains should ideally be maintained by $C_f(t,\omega)$. Two of these properties respectively correspond to the instantaneous power and energy density given in (2-2) and (2-3). The remaining properties are summarized in Table 2-1 and discussed below. Properties 2 and 6, the finite support properties, are

$$f(t)=0 \text{ for } t\leq t_0 \text{ and/or } t\geq t_1$$

$$\Rightarrow C_f(t,\omega;\phi)=0 \text{ for } t\leq t_0 \text{ and/or } t\geq t_1 \qquad (2\text{-}7)$$

and

$$F(\omega)=0 \text{ for } \omega\leq\omega_0 \text{ and/or } \omega\geq\omega_1$$

$$\Rightarrow C_f(t,\omega;\phi)=0 \text{ for } \omega\leq\omega_0 \text{ and/or } \omega\geq\omega_1. \qquad (2\text{-}8)$$

They state that if the signal is time or bandlimited, its joint time-frequency representation should also be time or bandlimited to within the same region. Properties 3 and 7 state that a shift of the signal in time or frequency results in a corresponding shift in the time or frequency

Table 2-1. Characteristic signal properties.

Time Domain Properties:
1. instantaneous power 2. finite support 3. shifting 4. instantaneous frequency (analytic signals)
Frequency Domain Properties:
5. energy density 6. finite support 7. shifting 8. group delay
Signal Distribution:
9. real-valued 10. positivity

of $C_f(t,\omega)$ as follows:

$$g(t)=f(t-t_0) \Rightarrow C_g(t,\omega;\phi)=C_f(t-t_0,\omega;\phi); \qquad (2\text{-}9)$$

and

$$G(\omega)=F(\omega-\omega_0) \Rightarrow C_g(t,\omega;\phi)=C_f(t,\omega-\omega_0;\phi). \qquad (2\text{-}10)$$

Property 4 states that the average frequency of $C_f(t,\omega)$ at any time is equal to the derivative of the phase of the signal, or

$$\frac{\int_{-\infty}^{\infty}\omega C_f(t,\omega;\phi)d\omega}{\int_{-\infty}^{\infty}C_f(t,\omega;\phi)d\omega} = -d\theta(t)/dt, \qquad (2\text{-}11)$$

where the envelope of f(t) is of the form $a(t)e^{j\theta(t)}$. For real signals, the average frequency is zero and for analytic signals, the average frequency is equal to the instantaneous frequency. Property 8 states that the average time of $C_f(t,\omega)$ is equal to the group delay. That

is, if $\tau_g(\omega)=-d\phi(\omega)/d\omega$, where $F(\omega)=R(\omega)e^{j\phi(\omega)}$, then

$$\frac{\int_{-\infty}^{\infty} tC_f(t,\omega;\phi)dt}{\int_{-\infty}^{\infty} C_f(t,\omega;\phi)dt} = \tau_g(\omega). \qquad (2\text{-}12)$$

Property 9,

$$C_f(t,\omega;\phi) = C^*_f(t,\omega;\phi), \qquad (2\text{-}13)$$

states that $C_f(t,\omega)$ is real-valued. Property 10,

$$C_f(t,\omega;\phi)\geq 0, \qquad (2\text{-}14)$$

states that $C_f(t,\omega)$ is everywhere positive.

Although ideal, a generalized time-frequency representation for which all the properties in Table 2-1 hold, reguardless of the signal, is not possible. Consider, for example, a representation that satisfies properties 2 and 10: then from (2-7), if the signal is 0 for $t\leq 0$, $C_f(t,\omega)=0$ for $t\leq 0$. By definition, the time average of $C_f(t,\omega)$, where Property 10 defines $C_f(t,\omega)$ as everwhere ≥ 0, is ≥ 0 at all frequencies. Obviously,

Property 8 (Equation 2-12) in this case does not hold as the group delay for causal signals is not strictly positive. Although these properties may not absolutely hold, they may be satisfactorily maintained under certain signal constraints.

Again, consider the case where $C_f(t,\omega)$ corresponds to the STSD. Integrating the STSD over time at an arbitrarily fixed frequency was shown to produce a windowed average of the energy density of the form (2-6). In suit, the first moment over time of $S(t,\omega)$ (where for simplicity the subscript has been omitted) produces an averaging of the group delay. That is

$$\frac{\int_{-\infty}^{\infty} tS(t,\omega)dt}{\int_{-\infty}^{\infty} S(t,\omega)dt} = \frac{\int_{-\infty}^{\infty} t_g(\zeta)F'(\omega,\zeta)d\zeta}{\int_{-\infty}^{\infty} F'(\omega,\zeta)d\zeta},$$

where

$$F'(\omega,\zeta) = |F(\zeta)|^2|H(\omega-\zeta)|^2. \qquad (2\text{-}15)$$

If the significant spectral contributions of the signal do not change appreciably over the length of the window, then (2-15) and (2-16) correspondingly do not vary greatly over the window length and the respective average values are sufficiently close to the instantaneous values. A signal

of this type is often referred to as quasi-stationary or stationary over the window. Conversely, a signal whose dispersion characteristics undergo significant change over the window length is referred to as a time-varying or nonstationary signal. Naturally, the STSD would not be an adequate representation for signals that are time-varying and an alternative choice for $C_f(t,\omega)$ must be considered. More specifically, the objective in selecting $C_f(t,\omega)$ is to optimally represent signal concentration over time and frequency, which is, in fact, limited by the uncertainty relation of Heisenberge, and the Cohen class with its ideal set of properties serve as a viable basis on which to relate time-frequency representations of a signal [6,7,8,11].

2.2 Wigner Distribution

The continuous Wigner distribution (WD) of a signal f(t) is defined as

$$W(t,\omega) = \int_{-\infty}^{\infty} f(t+\tau/2)f^*(t-\tau/2)e^{-j\omega\tau}d\tau \qquad (2\text{-}16)$$

and is obtained from (2-1) by setting ϕ=1. The WD satisfies all of the defining properties in Table 2-1 except that of positivity [6]. (Note should be made that

Property 6 is constrained to a single bandpass region; the case of multiple nonzero bands is discussed in Section 3.3.) The occurrance of negative values that allows the other nine properties to simultaneously hold, however, prevents the WD fron being described formally as a distribution. The existence of the negative values may be equated to that of negative energy states introduced in quantum mechanics or of negative temperatures in thermodynamics.

Hudson [23] has proven the WD as applied to quantum mechanics, in the form of a phase-space distribution, to be everywhere positive if and only if the Schrodinger state vector is the exponent of a quadratic. With respect to signal estimation, Hudson's proof equates to positivity for the WD given the signal envelope is the exponent of a quadratic. In an earlier paper, DeBruijn proved application of constrained Gaussian smoothing operators on the WD could effectively elliminate the negative value problem [5].

In addition to those properties which characterize the Cohen class, the WD has a number of desirable attributes that make it well suited to signal/image processing applications [6,14,15]. For example, two common linear operations on signals are filtering and modulation. The duality of each operation with respect to time and frequency is evident in the resulting WD. More specifically,

filtering: (2-17)

$$y(t) = x(t) *_c h(t) \Rightarrow W_y(t,\omega) = \int_{-\infty}^{\infty} W_x(t-\tau,\omega) W_h(t,\omega) d\tau$$

modulation: (2-18)

$$y(t) = x(t)h(t) \Rightarrow W_y(t,\omega) = \int_{-\infty}^{\infty} W_x(t,\omega) W_h(t,\omega-v) dv$$

where W_y, W_x, W_h, are the respective WD's of y, x, h and $*_c$ denotes convolution. These two operations are important to real-time processing of the WD. Another characteristic of the Wigner is

$$W_f(t,\omega) = W_F(\omega,-t), \tag{2-19}$$

where W_f is the WD of the signal f(t), as defined in Equation (2-16) and W_F is the WD of the signal transform with integration over frequency as follows,

$$W_F(\omega,t) = \int_{-\infty}^{\infty} F(\omega+\xi/2) F^*(\omega-\xi/2) e^{-j\omega\tau} d\xi, \tag{2-20}$$

where $F(\omega)$ denotes the Fourier transform of f(t). The

duality between t and ω is evident in (2-19). This demonstrates again the validity of the WD as a time-frequency representation.

2.2.1 Windowed Wigner Distribution

In practical applications, a windowed version of the WD is used. If the window is of an appropriate Gaussian form the WD becomes observably nonnegative (Section 2.2.3). The WD of a windowed version of the signal follows from (2-18). If h(t) denotes a window, centered at t', applied to the signal, f(t), such that y(t)=f(t)h(t-t'), then

$$W_y(t,t',\omega) = (1/2\pi)\int_{-\infty}^{\infty} W_x(t,v)W_h(t-t',\omega-v)dv. \quad (2\text{-}21)$$

The WD can be seen as a function of the window position. Sliding the window along the time axis yields a series of WDs. The pseudo-WD, or pWD, is defined by observing the WD only at the window center t=t'. As the window moves along the time axis, it is expressed as

$$W_y(t,\omega) = (1/2\pi)\int_{-\infty}^{\infty} W_x(t,v)W_h(0,\omega-v)dv, \quad (2\text{-}22)$$

or, equivalently,

$$W(t,\omega) = \int_{-\infty}^{\infty} f(t+\tau/2)f^*(t-\tau/2)h(\tau/2)h^*(-\tau/2)e^{-j\omega\tau}d\tau. \quad (2-23)$$

The pWD produces spreading, or smoothing, in frequency that is equivalent to a convolution in the frequency domain of the WD of the signal and the WD (at t=0) of the window. Although the time domain properties in Table 2-1 are unaffected, the frequency smoothing translates to an averaging over the WD of the window for properties 5, 6 and 8. For practical applications there is no loss of generality in assuming the window function to be real-valued, in which case Property 4 will be maintained precisely by the pWD. For a causal system with impulse response approaching zero outside a finite time, T_0, (as for example loudspeakers [8]) the pWD closely resembles the WD if the window length exceeds T_0 (i.e. properties 5,6 and 8 almost hold).

2.2.2 Doubly Weighted Wigner Distribution

Frequency spreading of the pWD relative to each time slice can be reduced via a moving filter. That is, using the t-ω duality of the WD as given in (2-19), each filtered pWD slice is satisfied by (2-17) where the convolution

function W_h becomes the WD of the filter at time t=0. In this form the spWD can be treated as a doubly biased(or smoothed) WD where the relative frequency and time resolutions can be controlled by the choice of window and filter. This is the one-dimensional counterpart to the composite WD in [27].

To compute the weighted WD, the signal is windowed over a finite time centered about an arbitrarily fixed time t_0, producing $f(t)=f(t)h_1(t-t_0)$. The transform of the windowed function is computed then multiplied by a filter centered about a frequency ω_0, to yield

$$F_{t0,\omega0}(\omega) = \{F(\omega)*H_1(\omega)\exp(-j\omega t_0)\}H_2(\omega-\omega_0) \quad (2-24)$$

or, equivalently,

$$f_{t0,\omega0}(t) = \{f(t)h_1(t-t_0)\}*h_2(t)\exp(j\omega_0 t). \quad (2-25)$$

The weighted WD is defined by observing the WD at $t=t_0$ and $\omega=\omega_0$ and at each point (t,ω) expresses as

$$W_f(t,\omega) = (1/2\pi)\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} W_f(z,v)W_{h1}(0,\omega-v)W_{h2}(t-z,0)dzdv. \quad (2-26)$$

As the equivalent of windowing and filtering is performed on the signal to produce the wWD, a weighted average of both the time and frequency domain signal properties is maintained by the wWD. In fact, all other members of the Cohen class can similarily be represented by weighted averages of the WD in the following manner [6,14],

$$C_f(t,\omega;\phi) = (1/2\pi)\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} W_f(\zeta,\xi)\theta(t-\zeta,\omega-\xi)d\zeta d\xi, \qquad (2\text{-}27)$$

for $W_f(t,\omega)$ the WD of f(t) and θ the appropriate weighting where

$$\theta(t,\omega) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\phi(\zeta,\xi)e^{j(\omega\zeta-\xi t)}d\zeta d\xi, \qquad (2\text{-}28)$$

for $\phi(t,\omega)$ satisfying (2-1). For example, setting $\theta(t,\omega)=e^{j(\omega\zeta-\xi t)}$ in (2-27) would yield the double FT of $W_f(t,\omega)$, which would in fact produce the ambiguity function of the signal (Section 2.3).

2.2.3 Maximum Concentration Weighting Functions

A natural question arises as to the optimum selection of window and filter functions. As known, signal concentration in the t-ω plane is limited by Heisenberg's uncertainty inequality [5]

$$[\int_{-\infty}^{\infty} t^2 |f(t)|^2 dt]^{1/2} [\int_{-\infty}^{\infty} \omega^2 F(\omega) d\omega]^{1/2} \geq ||f||/4\pi \qquad (2\text{-}29)$$

where $||\cdot||$ denotes the L_2 norm and the lobe center is assumed to be at the origin. For the case of the WD (properties 1 and 5) Equation (2-29) translates to

$$\int_{-\infty}^{\infty} t^2 W(t,\omega) dt d\omega \int_{-\infty}^{\infty} \omega^2 W(t,\omega) dt d\omega \geq (||f||/4\pi)^2 \qquad (2\text{-}30)$$

Both second global moments are strictly positive for the WD [14] and, therefore, (2-30) can be interpreted as the product of the variances of $W(t,\omega)$ about the lobe center over time and over frequency. A well known fact from quantum mechanics is that the inequality in (2-29) becomes an equality if the signal waveform is a Gaussian (i.e. the measure of uncertainty is a minimum). Furthermore, Heisenberg's inequality approaches an equality if the

waveform uniformly approaches a function for which it becomes an equality [5]. Therefore, a logical conclusion would be to select the window to approach a Gaussian shape. From (2-19) and (2-20) the same conclusion holds for the selection of the moving filter.

From (2-26) the weighting function (or moving average) in (2-27) for the weighted WD is a separable function of the form $\theta_1(t)\theta_2(\omega)$. From (2-16) and (2-26), if the window is a Gaussian with variance β_w, the corresponding smoothing function, $\theta_2(\omega)$, is also a Gaussian, with variance $2/\beta_w$; likewise, from (2-19), (2-20) and (2-26), if the filter is a Gaussian with variance β_f, the smoothing function $\theta_1(t)$ is a Gaussian with variance $2/\beta_f$. For the case of Gaussian windowing and Gaussian filtering on the signal then the wWD satisfies (2-27) where

$$\theta(t,\omega)=\pi(\beta_w\beta_f)^{1/2}\exp[-\beta_f t^2/4-\beta_w\omega^2/4] \qquad (2\text{-}31)$$

which is the Weierstrass transform and converges rapidly. Furthermore, if the window and filter are selected such that $\beta_w\beta_f<16$, then the corresponding smoothing functions satisfy the positivity requirement established by DeBruijn [5] and the weighted WD becomes everywhere positive.

2.3 Ambiguity Function

The double Fourier transform of the WD is the ambiguity function (AF), which is common to radar and sonar applications. The AF expresses as

$$A(t,\omega) = \int_{-\infty}^{\infty} f(\tau+t/2)f^*(\tau-t/2)e^{-j\omega\tau}d\tau \qquad (2\text{-}32)$$

and can be obtained from (2-1) by setting $\phi=2\pi\delta(\tau-t)\delta(\xi-\omega)$. Certain properties of the AF prohibit ideal resolution between targets. Specifically, these properties require that the maximum height and the total volume of the AF are both equal to $(2E)^2$ where E is the energy in the doppler-shifted signal. In effect a finite height with finite volume is used to represent the ambiguity diagram possibly preventing resolution between two closely spaced targets. The width of the peak may be narrowed; however, the height of the AF must be raised elsewhere to retain the $(2E)^2$ volume causing unwanted peaks (pillow effect) [33]. As such a trade-off between doppler frequency resolution and range ambiguity must be made.

The ambiguity function has been applied extensively to both radar and sonar. Its plot, called the ambiguity diagram, serves to represent the response of the matched filter to the signal to which it is matched and to

doppler-frequency-shifted signals. As such, the AF can be used as an indicator of the limitations and advantages of particular waveform classes. The AF is a representation of time delay and frequency shift and its properties should be interpreted accordingly. Consider, for example, the shifting properties in Table 2-1. Time delay and frequency translation of the signal directly correspond to respective time and frequency shifts in the WD. In the AF, time delay of the signal manifests as a translation in frequency and frequency translation appears in the AF plane as a shift in time. This clearly agrees with the double Fourier transform relationship between the WD and the AF.

The Fourier transform relationship between the AF and the WD can be used to translate the wealth of knowledge associated with the AF to the WD, keeping in mind the transform relationship. Consider the following example, a Gaussian waveform is known to produce maximum ambiguity (or variance) in the AF plane; the double Fourier transform relationship, then, would indicate the Gaussian to produce minimum variance for the WD, which, in fact, is the case.

2.4 Short-time Spectral Density

The short-time spectral density (STSD), defined in (2-4) and (2-5), can be obtained from (2-1) by setting ϕ equal to the AF of the window, h(t), and satisfies only properties 3,7,9 and 10. As discussed in Section 2.1, if

the signal properties do not vary greatly over the interval in which the spectrogram is being taken, the remaining properties are suitably averaged; this is generally true for short-time stationary signals where the signal is assumed stationary over short time intervals and the STFT is computed over the interval then concatenated in time to produce the spectrum. For signals characterized by instationarities in the signal, which is true for audio signals (hence speech) [24], this quasi-stationary assumtion can mask important information contained in the signal.

A comparison can now be made between the spectrogram expressed in this manner and the smoothed-psuedo-WD; for the spectrogram smoothing in both the time and frequency domains are interdependent leading to the well-known problem of trade-off between spectral resolution and the quasi-stationary assumption of the signal. In the weightedWD this problem is avoided by allowing time and frequency smoothing to be computed independently, that is $\theta(t,\omega)$ in (2-27) is a separable function of the form $\theta_1(t)\theta_2(\omega)$ for the wWD.

Another common two-dimensional representation for audio signals is the cumulative spectra (CS). The cummulative spectra is actually a special type of spectrogram where the only properties in Table 2-1 satisfied by the CS are also 3,7,9 and 10.

CHAPTER 3
APPLICATIONS TO DIGITAL SIGNAL PROCESSING

3.1 Discrete Wigner Distribution

Generally estimation of a time-varying spectra is based on the short-time Fourier transform (STFT). Here the process is assumed stationary over short time intervals and the discrete Fourier transforms (DFTs) of these intervals are ensembled then used to characterize the spectrum. STFT methods postulate stationarity over the observation interval; this defines a compromise between frequency resolution and adequate preservation of the nonstationarities in the signal dispersion characteristics. Despite optimum windowing attempts, STFT techniques generally produce a representation that is overly smoothed in both time and frequency. Still the computational efficiency of FFT data processing makes STFT techniques attractive for real-time implementation and the short-time periodogram (STP), defined as the square modulus of the STFT, is a common tool used to characterize time-varying spectra. The discrete form of the WD, the DWD, uses an FFT

and serves as a viable alternative to conventional STFT methods (namely the STP and modified (or smoothed)-STP). In this section the DWD will be developed. Because the DWD is a competative alternative to the STP, a detailed comparison between each will be made after the DWD has been fully developed.

The discrete time Wigner distribution (DTWD) is defined as

$$W(n,\omega)=2\sum_{m=-\infty}^{\infty} x(n+m)x^*(n-m)e^{-j2\omega m} \qquad (3\text{-}1)$$

where, as is common practice, n designates nT, for T the time between samples. By observing the "power of two" of the exponent, the DTWD can be seen to be periodic in frequency with period π/T (whereas the Fourier spectrum is periodic with period $2\pi/T$) which can lead to aliasing of the DTWD in the frequency domain. An alternative definition is possible which omits the power of two in the exponent. However, by doing this one must recall that frequencies at ω in the signal are translated out to nearly 2ω during computation of the inner product function, or WD kernel, $x(n+m)x^*(n-m)$. For clarity, consider a sinusoid with frequency f_0=375Hz (Figure 3-1): in the inner product

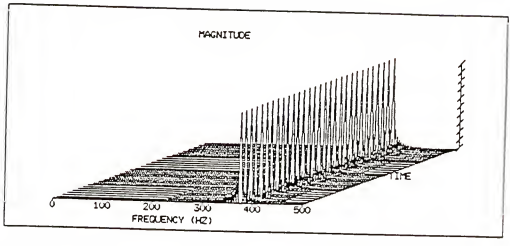

Figure 3-1. STFT of a single sinusoid at f_0=375Hz and sampled at 1kHZ.

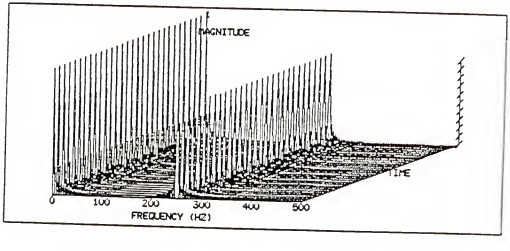

Figure 3-2. DWD of a single sinusoid at f_0=375Hz sampled at 1kHz where the tone at $2\omega_0$ is an aliased spectral component.

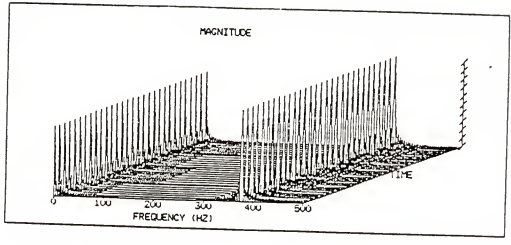

Figure 3-3. DWD of the 375Hz sinusoid sampled at 2kHz.

f_0 is mapped to dc and $2f_0$, which folds into the t-f plane interval as an aliased contribution (Figure 3-2). (Notice dc information is difficult to obtain from the WD due to the spectral "corruption" around dc associated with the inner product.) To prevent aliasing in the frequency domain, the signal must be sampled at a rate commensurate with the effective bandwidth of the WD kernel. This can be guaranteed by sampling the signal at double the Nyquist rate.

By defining the DTWD as in (3-1), frequencies that occur at ω in the signal also occur at ω in the DTWD. That is, frequencies translated out to 2ω during computation of the inner product are isomorphically folded back to ω (Figure 3-3). If the signal has been sampled at twice the Nyquist rate the DTWD for each period is zero over the second half of the interval, $\pi/T<\omega\leq 2\pi/T$; hence no frequency aliasing occurs. If supplied with a discrete time sequence sampled at the Nyquist rate, interpolation of the data can be used to satisfy this criterion. Observe from Figures 3-2 and 3-3, the DWD introduces spectral components around dc. Where this inherent dc "corruption" derives from and its consequential elimination are discussed in Section 3.3.

Alternative definitions for the DTWD have been considered [18,22,34], each of particular merit; however, the need to sample at twice the Nyquist rate to combat the problem of aliasing has been common to all. That is, the

DTWD of the oversampled signal is fully described over $(0,\pi/T)$. Using the definition in (3-1) and assuming the signals to be sampled at twice the Nyquist rate, the Cohen class properties in Table 2-1 can be applied to the DTWD. Properties 1 and 5, the instantaneous power and energy density, defined by (2-2) and (2-3), respectively, in Chapter 2, express as

$$\int_{-\infty}^{\infty} W(n,\omega)d\omega = |x(n)|^2 \qquad (3\text{-}2)$$

and

$$\sum_{n=-\infty}^{\infty} W(n,\omega)=(|F(\omega)|^2+|F(\omega+\pi/T)|)^2, \qquad (3\text{-}3)$$

where the second term in (3-3) vanishes for the cases of analytic or oversampled signals. The finite support property with respect to time (2) follows in a straightforward manner from (2-7) and expresses as

$$x(n)=0 \quad n\geq n_0 \text{ and } n\leq n_1$$

$$\Rightarrow [W(n,\omega)]\text{mod}N=0, \quad n\geq n_0 \text{ and } n\leq n_1. \qquad (3\text{-}4)$$

The finite support property with respect to frequency (Property 6, Equation (2-8) is not as straightforward: for analytic signals, where the spectrum vanishes between π/T and $2\pi/T$, the following relation holds,

$$X(\omega)=0 \quad \omega_1<\omega<\pi/T$$

$$\Rightarrow[W(n,\omega)]\bmod\pi/T=0 \quad \omega_1<\pi/T; \tag{3-5}$$

and, for oversampled signals, where the spectrum vanishes between $\pi/2T$ and $3\pi/2T$, the following relation is satisfied,

$$X(\omega)=0 \quad \omega_1<\omega<\omega_2, \quad \omega_1<\pi/2T, \quad \omega_2>3\pi/2T$$

$$\Rightarrow W(n,\omega)\bmod\pi/T=0 \quad \omega_1<\omega<\omega_2-\pi/T. \tag{3-6}$$

As in the continuous Wigner, (3-5) and (3-6) exclude the case of multiple bands, which will be considered separately in Section 3.3. The shifting properties 3 and 7 (Equations (2-9) and (2-10)) correspond directly to the DTWD respectively as follows:

$$x(n-n_0)\Rightarrow W(n-n_0,\omega); \tag{3-7}$$

and

$$X(\omega-\omega_0) \Rightarrow W(n,\omega-\omega_0). \tag{3-8}$$

The group delay of the discrete-time sequence (Property 4) follows analogously from (2-12) as below

$$\tau_g(\omega) = \frac{\sum_{n=0}^{N-1} nW(n,\omega)}{\sum_{n=0}^{N-1} W(n,\omega)}. \tag{3-9}$$

The notion of an instantaneous frequency (Property 8) obviously cannot apply to discrete-time sequences; however, the instantaneous frequency defined for continuous-time sequences and satisfying (2-11) can be translated to the case of the DTWD by observing the forward and backward differences of the phase about n [15]. Here, the following equality holds

$$\left[\frac{\theta(n+1)-\theta(n-1)}{2}\right]_{\bmod \pi/T} = \frac{\int_{-\pi/2T}^{\pi/2T} \omega W(n,\omega)\,d\omega}{\int_{-\pi/2T}^{\pi/2T} W(n,\omega)\,d\omega}. \tag{3-10}$$

The real-valued property (9) of Equation (2-13) expresses for the DTWD as below,

$$W(n,\omega) \Rightarrow W^{*}(n,\omega). \tag{3-11}$$

As with the continuous case, property 10 (Equation (2-14)), that of positivity does not hold for the DTWD.

For finite duration sequences, of length NT, the DTWD can uniquely be expressed by its samples taken at intervals of $\omega = k\pi/NT$ as below,

$$W(n,k) = 2\sum_{m=-N/2+1}^{N/2} x(n+m)x^{*}(n-m)e^{-jmk2\pi/N}, \tag{3-12}$$

which is periodic in $\omega=\pi/T$. For analytic signals where the even or odd samples fully describe the signal, hence making the spectrum nonzero over only one-half the interval (i.e. single-sideband SSB), the signal need only be sampled at the Nyquist rate to prevent aliasing.

At this stage an important point needs to be made. Although aliasing in the frequency domain can be prevented by sampling the signal at a rate commensurate with the effective bandwidth of the Wigner kernel, aliasing in the DWD plane is unavoidable. This fact becomes clear by

recalling the inverse relation $\Delta f \sim 1/\Delta t$ discussed at the beginning of Chapter 2. A simple procedure to minimize aliasing while maintaining the relative simplicity of computing the DWD is to oversample the signal as stated, window the signal with a Gaussian shaped window to minimize the signal BT product (this will make the DWD observably nonnegative as well) and zero pad the window tails to reduce aliasing in time (or, equivalently, select the window variance to be relatively small). For a fixed transform length, windowing and zero padding will reduce frequency resolution at each time slice. Recall, however, for the DWD increasing the window length to improve frequency resolution does not degrade the temporal characteristics of the DWD. Therefore, the amount of zero padding is limited primarily by the processor capabilities.

3.2 Windowed Discrete Wigner Distribution

For real-time applications data sequences are frequently long in which case the pseudo-DWD is computed as

$$W(n,k)=2\sum_{m=-L+1}^{L-1} x(n+m)x^*(n-m)|h(m)|^2 \exp^{-j2\pi mk/2L}, \quad (3\text{-}13)$$

where h(m) is a window of length 2L-1 centered about n. The pseudo-DWD is usually called simply the DWD and, unless

stated otherwise, DWD will refer to the pseudo-DWD of (3-13). The pDWD is defined analogous to its continuous domain counterpart, the pWD and is, by definition, the concatenation of DWD slices over time where each slice, taken with respect to the window center, expresses as

$$W_n(0,k)=2\sum_{m=-L+1}^{L-1} x(m)x^*(-m)|h(m)|^2\exp^{-j2\pi mk/2L} \qquad (3\text{-}14)$$

and n designates the absolute center position of the moving window. Each DWD slice given above can be shown to be the DFT of a rearranged inner product function, $i_n^r(m)$, frequently referred to as the DWD kernel. The DWD kernel is obtained by rearranging the inner product, $x(m)h(m)x^*(-m)h^*(-m)$, such that terms in (3-14) from zero to N/2 form the first half of the DWD kernel and the terms in (3-14) from -N/2+1 to -1 form the second half of the kernel and $i_n^r(m)$ is defined to run from 0 to N-1. Using the relation $\exp(-jmk2\pi/N) = \exp(-j(m+N)k2\pi/N)$, the DWD can now be expressed as

$$W(n,k)=2\sum_{m=0}^{N-1} i_n^r(m)\exp^{-jmk2\pi/N} \qquad (3\text{-}15)$$

where $i_n^r(m)$ designates the DWD kernel as defined above taken with respect to a window centered about the point n. Expressed in the form of (3-15), the DWD is just the DFT of the kernel $i_n^r(m)$, hence lending itself readily to current FFT implementations.

Due to the symmetry of the DWD kernel, data rearrangement is not actually needed in the DWD computation if the signal is real. That is, the DFT of the windowed sequence by definition assumes periodicity of the signal outside the observation interval and application of the DWD kernel directly to the DFT, minus data rearrangement, is equivalent to starting the period at a different point in the periodic sequence. For practical applications there is no loss of generality in assuming the window to be a real symmetric function in which case the DWD expresses as

$$W(n,k)=2\sum_{m=0}^{N-1}|h(m)|^2 i_n(m)\exp^{-jmk2\pi/N} \qquad (3\text{-}16)$$

where $i_n(m)$ is the unrearranged DWD kernel given a rectangular window of length N and is by definition symmetric about n.

3.2.1 Comparison with the Short-time Periodogram

The DWD may now be compared directly with the popular short-time periodogram (STP). The STP, defined as the square modulus of the short-time Fourier transform, expresses as

$$P(n,k)=1/N\left|\sum_{m=0}^{N-1} x(m)h(n-m)e^{-j2\pi mk/N}\right|^2, \qquad (3\text{-}17)$$

where h(n) is a window of length N. Despite its limitations, the STP can easily be computed using FFTs and, as such, has been a standard estimation procedure. The primary limitation of the STP has been the trade-off between frequency resolution and preservation of signal instationarities in selecting the appropriate transform length. Consider for example a square pulse envelope on a 400Hz carrier as illustrated in Figure 3-4: in the STP (Figures 3-5 and 3-6) the lobe width increases at the edges where the quasi-stationary assumption over the window becomes invalid. For the longer transform length frequency smoothing is reduced, as evident by the narrowing of the spectral lobes; however, the transient part of the signal becomes significantly degraded. Application of a window as shown for the case of the Hamming window (Figures 3-7 and 3-8) can be used to improve the spectral output; however,

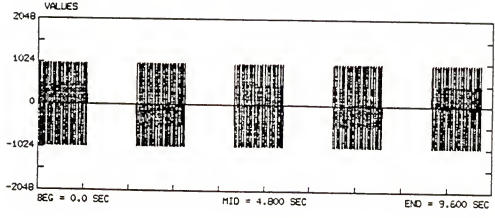

Figure 3-4. Square pulse envelope on a 400Hz carrier.

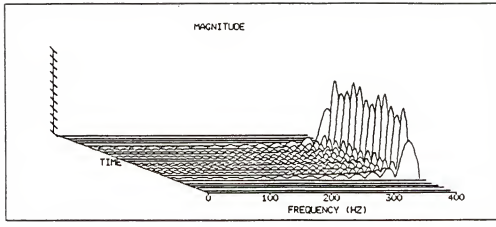

Figure 3-5. STP of square pulse modulation example for a 64 point transform length.

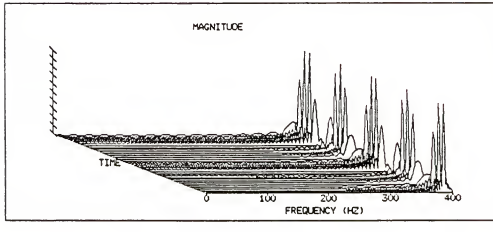

Figure 3-6. STP of square pulse modulation example for a 256 point transform length.

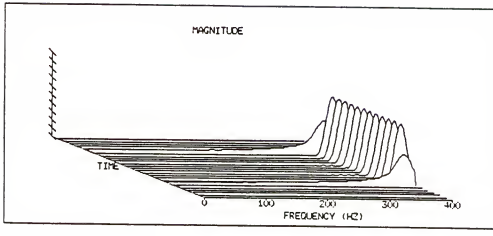

Figure 3-7. STP of square pulse modulation example for a 64 point transform length with Hamming windowing.

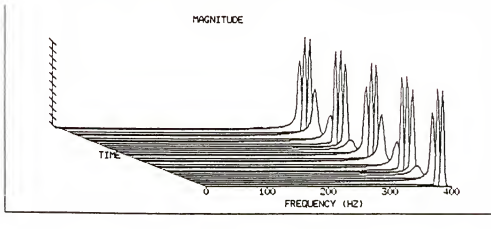

Figure 3-8. STP of square pulse modulation example for a 256 point transform length with Hamming windowing.

improvements are limited primarily to minor temporal changes.

The inherent t-ω trade-off limitation becomes apparent by observing the STP as the DSP analog to the STSD discussed in Chapter 2. Recall, the weighting function $\phi(t,\omega)$ in (2-1) was a nonseparable function for the STSD which assumed the form of the AF of the window. Using the double Fourier transform relation between ϕ and θ (Equation (2-28)) and between the AF and the WD, the STP can be expressed as a weighted average of the (unwindowed) DWD of (3-12) where the weighting term, $\theta(n,k)$, is a nonseparable function of the form

$$\theta_S(n,k) = \sum_{m=0}^{N-1} h(n+m)h(n-m)e^{-j2\pi mk/N}. \qquad (3-18)$$

For the pDWD the weighting function resulting from windowing assumes the following form

$$\theta_W(k) = \sum_{m=0}^{N-1} h(m)h(-m)e^{-j2\pi mk/N}, \qquad (3-19)$$

which is the DWD of the window at time zero (Equation (2-22)). Here the weighting term is a function of k only,

where windowing of the DWD does not smooth the time domain signal properties avoiding the standard time/frequency trade-off problem associated with the STP. Figures 3-9 and 3-10 give the DWD for respective 64 and 256 point transforms of the square pulse modulation in Figure 3-4. In the DWD the lobe widths remain relatively constant over the time interval. As with the STP, appropriate windowing of the data can be used to improve the estimation particularly for the case of Gaussian smoothing as discussed in Section 2.2 and shown in Figures 3-11 and 3-12. Because θ is not a function of n for the DWD, frequency resolution is limited primarily by the inherent windowing associated with the FFT. That is, frequency resolution is primarily limited by the capabilities of the FFT processor.

3.2.2 Comparative Examples

Comparison between the DWD and the STP can be best illustrated through examples. To establish a like comparison, equivalent transform lengths and rectangular windowing will be applied to both representations and respective magnitude and square magnitude plots for each will be given in the figures. The signals will be carried at lower frequencies than in practical systems. This is done for illustrative clarity and should not affect the purpose of the examples.

Figures 3-13 and 3-14 give the respective 64 point and 256 point STPs for a sine pulse on a 400Hz carrier (Figure 3-15). For the sine pulse there exists no sudden change in signal amplitude to severely degrade the STP and for the 64 point transform the quasi-stationary assumption appears to hold fairly well. Naturally higher frequency resolution is attained with the larger 256 point transform, however; the quasi-stationary assumption over the window does not hold as well, as becomes apparent in the temporal amplitude variation of the signal. Figures 3-16 and 3-17 are the corresponding 64 and 256 point DWDs of the sine pulse modulation.

Although the two examples above graphically illustrate the potential of the DWD for representing a signal over the $t-\omega$ plane, the importance of the DWD has been its application to time-varying signals or signals whose dispersion characteristics undergo significant change over time. An obvious example is the case of FM signals where concentration of the signal energy shifts over time. The larger the dispersion index the larger the variance of the corresponding spectral output. The dispersion index, β, is defined as the product of the period of the modulating signal and its amplitude [35] (i.e. the maximum frequency deviation from the carrier frequency of the FM signal, Δf). For narrowband signals β should be on the order of $\leq .2$; as β becomes large ($\rightarrow\infty$) the resolution band approaches the peak-to-peak deviation in the FM signal.

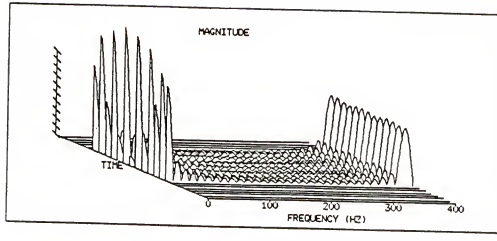

Figure 3-9. DWD of square pulse modulation example for a 64 point transform length.

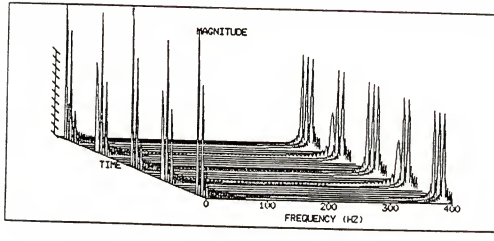

Figure 3-10. DWD of square pulse modulation example for a 256 point transform length.

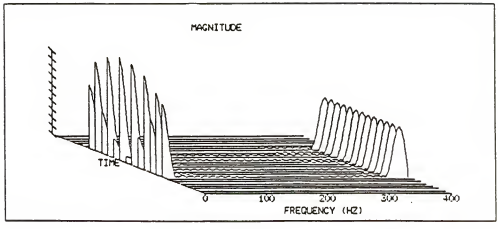

Figure 3-11. DWD of square pulse modulation example for a 64 point transform length with truncated Gaussian windowing.

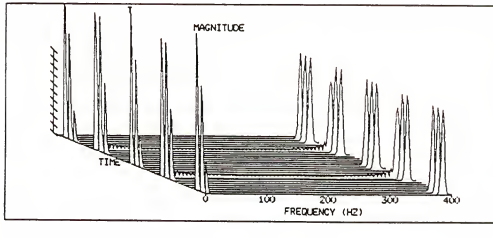

Figure 3-12. DWD of square pulse modulation example for a 256 point transform length with truncated Gaussian windowing.

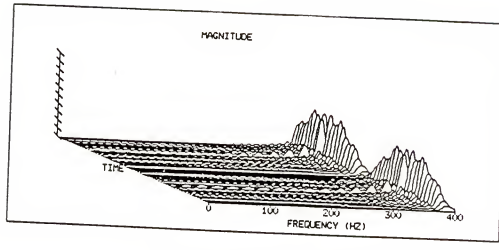

Figure 3-13. STP of sine pulse modulation example for N=64.

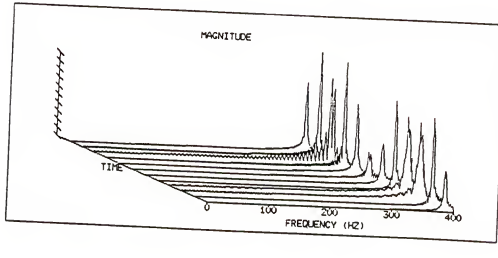

Figure 3-14. STP of sine pulse modulation example for N=256.

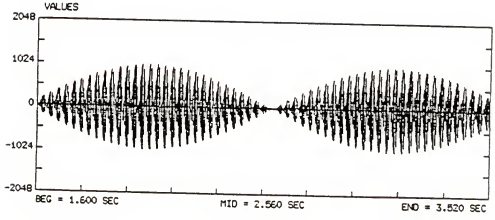

Figure 3-15. Sine pulse envelope on a 400Hz carrier.

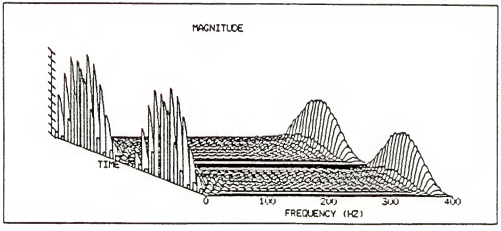

Figure 3-16. DWD of sine pulse modulation example for N=64.

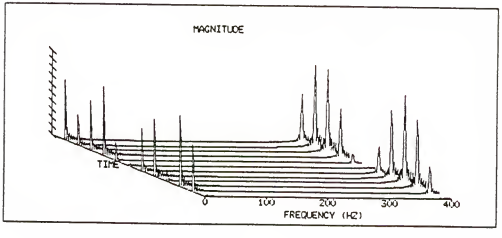

Figure 3-17. DWD of sine pulse modulation example for N=256.

A standard example of FM modulation is the case of sinusoidal FM on a carrier where the dispersion index equates to that of the modulation index. Figure 3-18 illustrates a sinusoidal FM signal on a carrier, for Δf=400Hz, over one-half the period of the modulating signal. The modulation index is large, β=800, and the signal becomes distorted at the peak points. For a modulation of this magnitude the spectral band approaches the peak-to-peak deviation. Figure 3-19 illustrates the STP of the sinusoidal FM. Application of the window at each time slice has effectively reduced the dispersion index (notice, the modulation index and the dispersion index are no longer equivalent); still the frequency band at each time slice is large. A smaller transform length will further narrow the band; however, will increase the effective bandwidth (i.e. reduce resolution). This translates to the well-known trade-off associated with the STP. That is, selection of the window length with respect to frequency resolution and assumption of stationarity. In the DWD kernel Δf is the maximum deviation, over the window, of the derivative of [m(t)-m(-t)], for m(t) the modulating signal, and β_{DWD} is effectively reduced to β_{STP}-(d/dt){max over T of m(T-t)} which implies $\beta_{DWD} << \beta_{STP}$. Figure 3-20 illustrates the superiority of the DWD in estimating the sinusoidal FM.

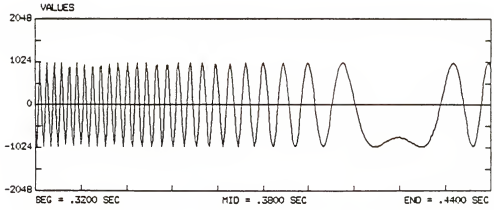

Figure 3-18. Sinusoidal FM signal for f_c=400Hz, β=800.

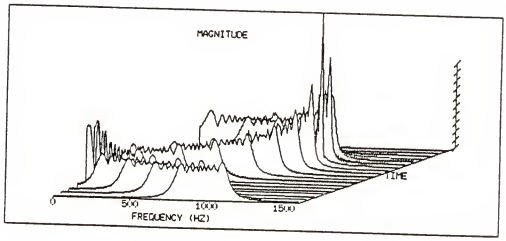

Figure 3-19. STP of sinusoidal FM example for N=256.

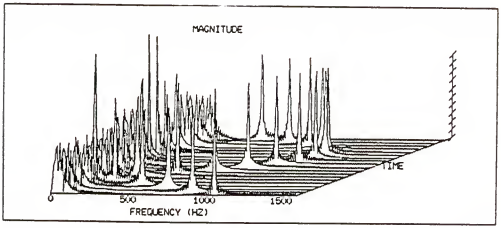

Figure 3-20. DWD of sinusoidal FM example for N=256.

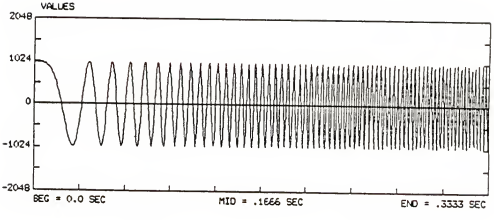

Figure 3-21. Cosine chirp signal.

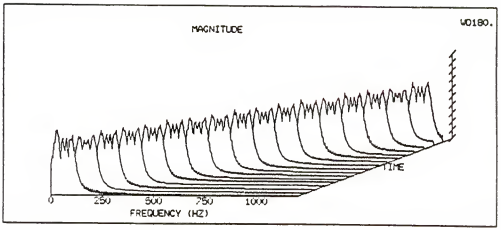

Figure 3-22. STP of cosine chirp example for N=256.

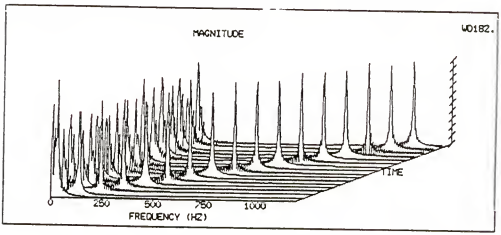

Figure 3-23. DWD of cosine chirp example for N=256.

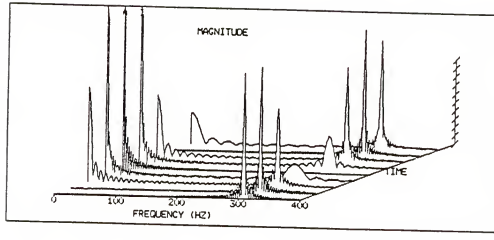

Figure 3-24. STP of square pulse FM example for N=256.

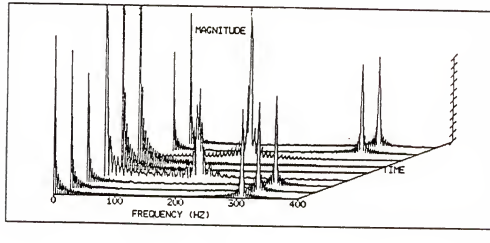

Figure 3-25. DWD of square pulse FM example for N=256.

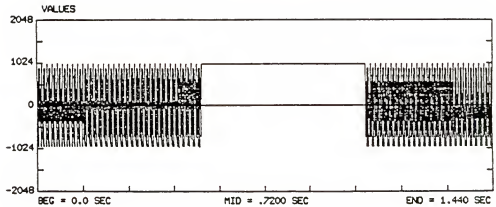

Figure 3-26. Square pulse FM signal for f_c=0, f_m=300Hz.

An important signal type to many applications is the cosine chirp (Figure 3-21) where the signal frequency varies linearly over time. Figures 3-22 and 3-23 are the respective 256 point STP and DWD of the cosine chirp. Even for the same transform length frequency resolution is poorer for the STP; this results from the postulation that Δf is small over the length of the window.

Figures 3-24 and 3-25 illustrate the comparison between the STP and DWD for square pulse FM (Figure 3-26). The greatest difference occurs at the point where the signal spectrum suddenly shifts: in the STP the quasi-stationary assumption obviously does not hold and primary lobes are present at both frequencies; in the DWD the points of transition are represented by an intermediate lobe to display the temporal variation of the spectrum.

3.3 Smoothed-pseudo Discrete Wigner Distribution

An important property of the WD, and equivalently the DWD, is its bilinear nature. That is, the DWD of the sum of two signals does not produce the sum of the two DWDs of the signals but the sum of the DWDs of the two signals plus a cross-DWD [6]. More explicitly, if $y(n)=x(n)+z(n)$ and $W_x(n,k)$ and $W_z(n,k)$ denote the respective DWDs of $x(n)$ and $z(n)$, then the DWD of $y(n)$ expresses as below

$$W_y(n,k)=W_x(n,k)+W_z(n,k)+4Re\{\sum_{m=0}^{N-1} x(n+m)z^*(n-m)e^{-j2\pi mk/N}\}. \quad (3-20)$$

The bilinear nature of the DWD is especially important to estimation of multi-component signals as the DWD produces cross-term components at each DWD slice. For example, consider a signal to be the sum of two sinusoids (Figure 3-27): the DWD of the signals produces the DWD of each sinusoidal component plus the cross-DWD of the two sinusoids which produces frequency components, called cross-terms, at dc and the sum and difference frequencies of the two sinusoids. Figures 3-28 and 3-29 are the respective STP and DWD of the composite 800 and 600Hz tones in Figure 3-27: the DWD is corrupted by the sum and difference frequencies which have been isomorphically mapped down by a factor of two due to oversampling of the signal. (Note should be made that (3-29) is a magnitude plot.)

The physical significance of these cross-term components is open to interpretation and further research. Chester [36] proposed a significance to the cross-term components for the case of speech signals by modelling the auditory properties of the ear as a receiver. For closely spaced tones the DWD cross-terms may be interpreted as a physical indication of the phase-coupling between the two tones which would invariably maintain a nonstationary characteristic over time.

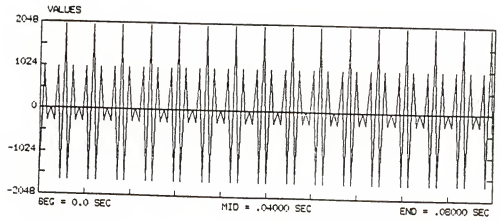

Figure 3-27. Sum of two sinusoids for f_1=600Hz, f_2=800Hz.

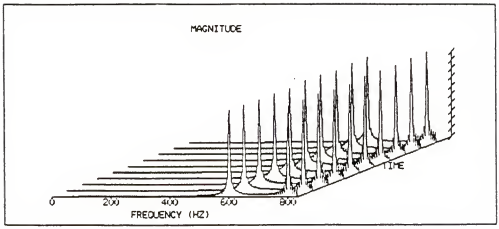

Figure 3-28. STP of composite sinusoid example for N=256.

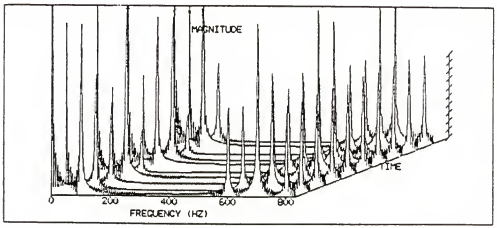

Figure 3-29. DWD, for N=256, of composite sinusoid example where additional cross-term components at $(f_1+f_2)/2$ and $(f_1-f_2)/2$ are present.

A viable argument as to the physical significance of these cross terms can be made by recalling from Chapter 2 that the Wigner distribution attains negative values and observing that these negative contributions occur exclusively in the cross-term components. To see this, consider the Fourier decomposition of the signal x(t) at time t as shown below

$$x(t) \sim \sum_m C_m e^{j\omega_m t}: \tag{3-21}$$

from its bilinear property the WD of (3-21) equates to the sum of the WDs of each component plus the sum of the cross-WDs as follows;

$$W(t,\omega) \sim \sum_m |C_m|^2 \delta(\omega-\omega_n) + \sum_{mn}\sum |C_m C_m^*| \cos[(\omega_n-\omega_m)t]\,\delta(\omega-(\omega_n+\omega_m)/2). \tag{3-22}$$

The first term in (3-22) is strictly positive whereas the second summation term, corresponding to the cross-term components, contains both negative and positive terms. Recall, in Section 2.2 these negative values were demonstrated as important to the simultaneous preservation of the characterizing time and frequency domain properties

of the signal. Accordingly, the existence of the cross-terms may prove important to certain signal types. For many applications, however, these cross-terms serve primarily to obfuscate the estimation procedure. An obvious example is the case of multipath radio signals (Sections 3.3.2 and 3.3.4).

Flandrin [9] introduced a smoothed-pseudoDWD to combat the cross-term problem of the DWD where smoothing was performed on the kernel as below,

$$spW(n,k)=2\sum_{m=0}^{N-1}|h(m)|^2 e^{-2j\pi mk/N}\sum_{m'=-M+1}^{M-1} g(m')i_n^r(m+m'), \qquad (3\text{-}23)$$

where g(m) constitutes smoothing in the time direction and h(n) in the frequency direction. Figure 3-30 illustrates the 256 point spDWD of the composite sine waves pictured in Figure 3-27 for M=32. The value of M is always smaller than N; its actual value is system and application dependent.

At this point two additional points concerning the bilinearity of the DWD should be mentioned. First, the dc corruption associated with the DWD of real signals may be treated as cross-term contribution between negative and positive frequency components. Application of the analytic signal can then serve to eliminate the dc corruption.

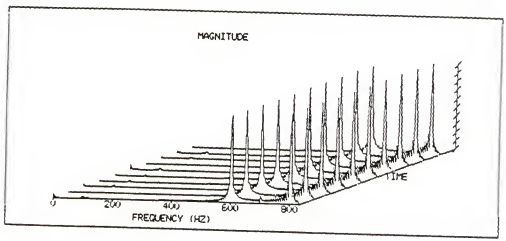

Figure 3-30. Smoothed-psuedo-DWD of composite sinusoid example for M=32 and N=256.

Second, from Equation (3-20) it follows the DWD is easily degraded when noise is present. That is, if z(n) represents additive noise then at each each DWD slice, n, the noise contribution comprises, in addition to the DWD of the noise, a cross-DWD of the signal with the noise defined by the DFT of $x_n(m)z_n^*(-m)+z_n(m)x_n^*(-m)$. For this reason the DWD has been treated as an estimator that requires clean signals [36]. This constraint severely limits the DWDs application potential. Fortunately the spDWD can be used to effect significant noise reduction.

The spDWD can be interpreted as the DSP analog to the weighted-WD of Section 2.2.2, where smoothing is effected on each pDWD slice as below

$$spW(n,k)=\sum_{m=-M+1}^{M-1} g(m)W(n+m,k). \qquad (3-24)$$

As with the continuous-time case, the spDWD can be interpreted as a weighted average of the DWD where the weighting function is separable and to preserve the moment properties of the DWD the weighting functions are real symmetric. From this interpretation follows that the spDWD weighting functions can be interpreted as filtering and modulation operations on the DWD as follows (Section 2.2.2 and 2.2.3)

$$spW(n,k) = \sum_{m=-M+1}^{M-1} \sum_{i=0}^{N-1} W(m,i)H'(k-i)g(n-m);$$

$$H'(k)=\sum_{n=0}^{N-1} h^2(n)e^{-j2\pi kn/N}. \qquad (3-25)$$

In this form the wealth of knowledge on filtering and modulation may be applied to the spDWD to achieve superior estimation for the two primary conditions under which the DWD produces poor results; multicomponent signals and noisy environments. Two primary advantages can be seen from the separability of the spDWD weighting function $\theta(n,k)=g(n)H'(k)$. First, the avoidance of time/frequency smoothing trade-off associated with the STP and second, the ability to effect smoothing similar to that performed on the STP (to improve performance) without the computational burden of computing multiple FFTs.

3.3.1 Examples of Cross-term Reduction

Consider for example the sum of two closely spaced cosine chirps, as pictured in Figure 3-31. The DWD (Figure 3-32) produces sufficient resolution between the two chirps but generates an additional cross-term component between the lobes at each time slice. Application of the spDWD (Figure 3-33) temporally smooths the signal which

effectively filters out the cross-term component as the distance between the signal lobes increases. For the case where the signal lobes are closely spaced a larger value of M is needed to effect a narrow band. However for time-varying signals, M must be small enough such that the convolution over which the signals inner product and the temporal weights are taken does not exceed the local time of stationarity; otherwise the nonstationary characteristics of the signal become obscured from too much temporal smoothing. Figure 3-34 is the spDWD of the dual cosine chirp for the case of smoothing over the entire window; although the cross-terms have been eliminated, temporal smoothing has degraded the signals time-varying characteristic and the spDWD representation approaches that of the STP (Figure 3-35) which postulates stationarity over the window.

The effect on the time-varying characteristic of a signal relative to the choice of M becomes apparent when the spDWD is applied to the pulse FM signal of Figure 3-21 for increasing values of M (Figures 3-36, 3-37, and 3-38). For M small (=4) the cross-term components are effectively eliminated. Recall, the dc corruption can be treated as cross-term components between negative and positive frequencies; therefore, if the spectral lobes are sufficiently away from dc, minimal smoothing will eliminate the dc terms. As M is increased the temporal

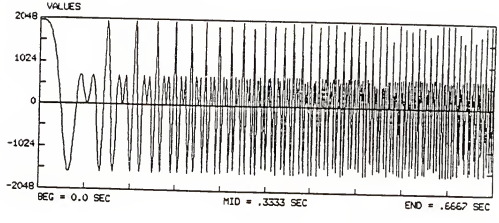

Figure 3-31. Dual cosine chirp.

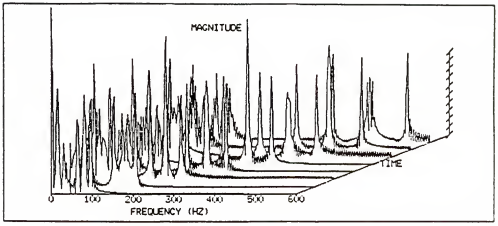

Figure 3-32. DWD of dual cosine chirp example for N=256.

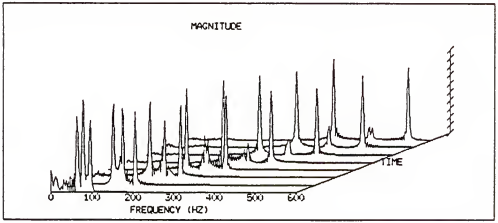

Figure 3-33. Smoothed-psuedo-DWD of dual cosine chirp for M=12 and N=256.

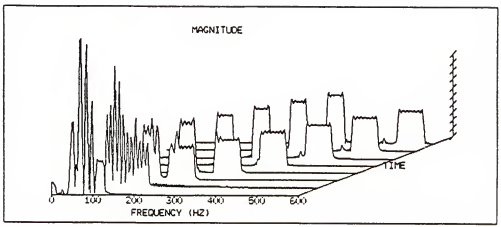

Figure 3-34. Smoothed-psuedo-DWD of dual cosine chirp for M=128 and N=256.

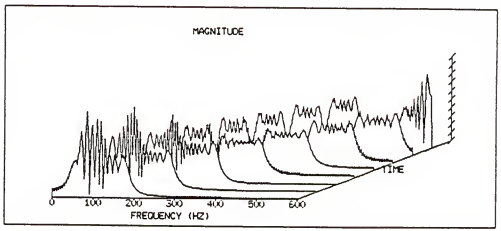

Figure 3-35. STP of dual cosine chirp.

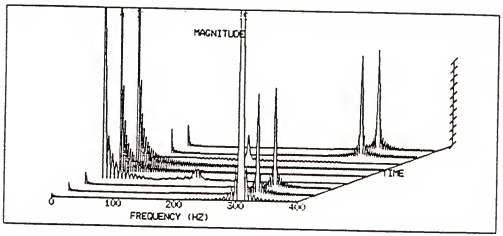

Figure 3-36. Smoothed-psuedo-DWD of square pulse FM example for M=4 and N=256.

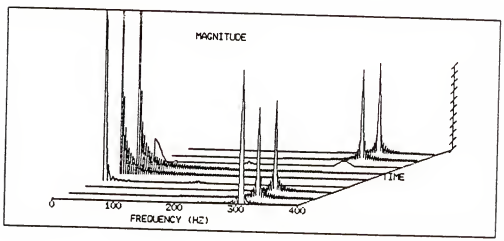

Figure 3-37. Smoothed-psuedo-DWD of square pulse FM example for M=32 and N=256.

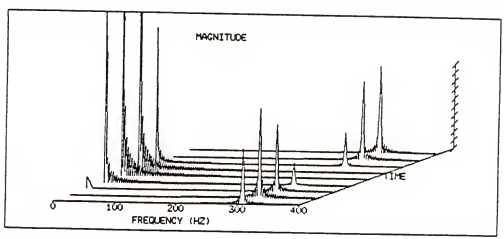

Figure 3-38. Smoothed-psuedo-DWD of square pulse FM example for M=128 and N=256.

characteristic of the spDWD approaches that of the STP, especially at the transient points where the local time of stationarity approaches zero (notice at these points any amount of smoothing exceeds the local time of stationarity for the pulse FM).

3.3.2 Multipath Radio Example

Figure 3-39 is a cosine chirp with an attenuated reflection delayed in time and shifted in frequency. A signal of this type is called a multipath signal and presents a common problem in radio transmission between low flying aircraft. The attenuated component results from ground reflection and interferes with the direct path signal. If the reflection contribution can be properly detected its presence may be accounted for at the receiver.

Figures 3-40 and 3-41 give the respective 256 point STP and DWD of the multipath signal in Figure 3-39. In the STP existence of the echo cannot be distinguished whereas in the DWD three separate tones appear: the tone corresponding to the original chirp the tone corresponding to the echo whose DWD is shifted in both frequency and time from the DWD of the original chirp and a cross-term between them. Application of the spDWD (Figure 3-42) can be used to eliminate that part of the cross-term that results from the two frequency components; however, the contribution to the cross-term component that results from the delay

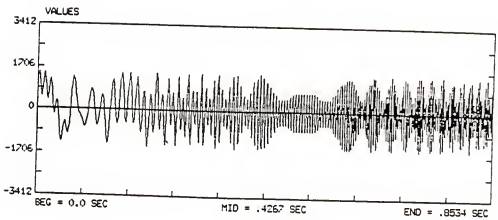

Figure 3-39. Multipath radio example: cosine chirp envelope with attenuated reflection delayed in time and shifted in frequency where τ_d=T/6 and f_d=200Hz.

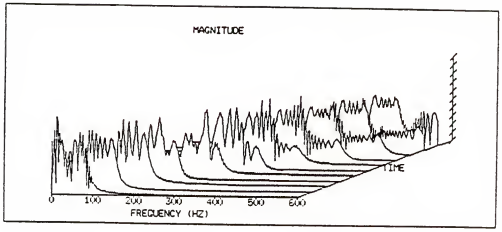

Figure 3-40. STP of multipath chirp example for N=256.

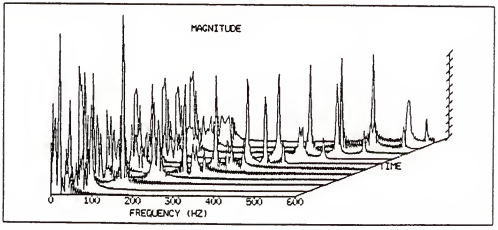

Figure 3-41. DWD of multipath chirp example for N=256.

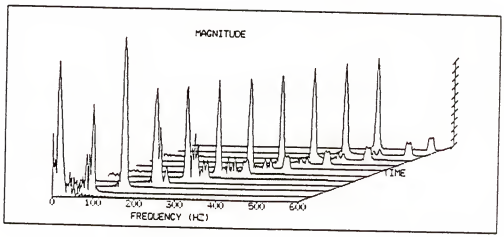

Figure 3-42. Smoothed-psuedo-DWD of multipath chirp example for M=32 and N=256.

between the chirps is relatively invariant to the smoothing procedure. That is, recall from Equation (3-25) the smoothing procedure effects a filtering operation on the signal at each time slice, or a moving filter. In addition, in the smoothing procedure, the shape of the delayed pulse becomes distorted as the spDWD performs temporal smoothing on the signal characteristics.

Consider for example the sum of a cosine chirp and a delayed version of the chirp with no frequency shift (Figure 3-43). In the DWD (Figure 3-44) three tones are present. The center lobe is a cross-term produced by the two outer lobes which result from the time displacement in the DWD of each chirp. Care must be taken not to confuse the time displaced lobes with frequency shifted lobes. The difference for the linear FM example is clear: in this example the displacement does not linearly increase over time as in the DWD of the dual chirp of Figure 3-32. Figure 3-45 is the spDWD for M=12. While smoothing eliminates the dc components that result from cross-terms between the negative and positive frequencies of the cosine chirp (Figure 3-33), the cross-term component between the time displaced chirps cannot be eliminated. Additional smoothing (Figure 3-46) does not reduce the relative amplitude of the cross-term but obscures the output by smoothing of the temporal signal properties. Figure 3-47 is the STP of the delayed chirp which precludes the existence of a delayed chirp altogether.

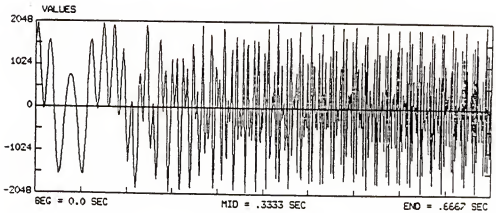

Figure 3-43. Chirp plus delay where τ_d=T/6.

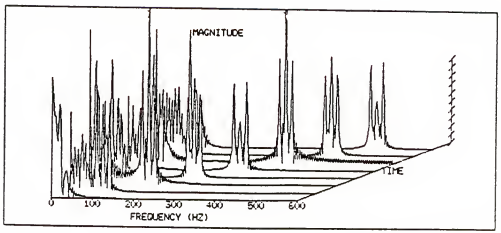

Figure 3-44. DWD of chirp plus delay example for N=256.

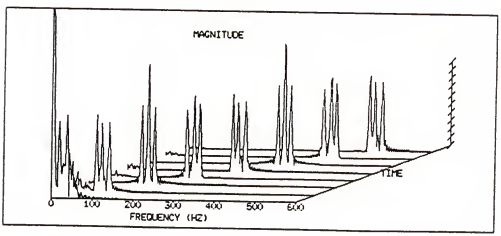

Figure 3-45. Smoothed-psuedo-DWD of chirp plus delay example for M=12 and N=256.

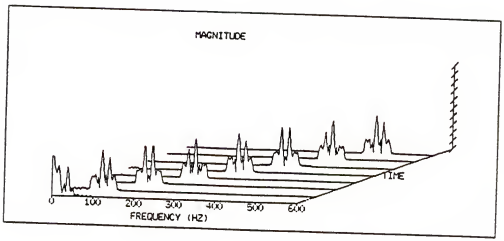

Figure 3-46. Smoothed-psuedo-DWD of chirp plus delay example for M=64 and N=256.

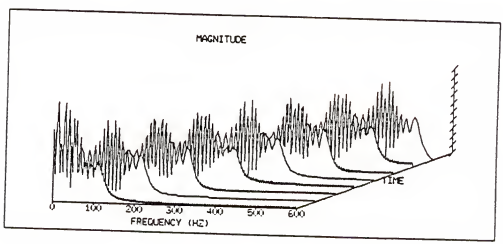

Figure 3-47. STP of chirp plus delay example for N=256.

Recall the double Fourier transform relationship between the DWD and the ambiguity function discussed in Chapter 2. In the ambiguity function instead of the signal waveform, the delay and doppler shift of the signal are ideally represented by location of a secondary lobe. (The classes of practical ambiguity diagrams and their limitations are well documented [33] and their inclusion is not necessary to this discussion.) A well-known property of the AF is that to obtain maximum resolution of the frequency and time delay while keeping the ambiguity which results from additional peaks to a minimum requires the bandwidth-time (BT) product of the signal be as large as possible. That is, the worst waveform, i.e. that which produces maximum variance in the AF plane, is the Gaussian. Following with the Fourier transform relationship between the AF and the DWD, the optimum window to minimize the variance in the DWD plane is a Gaussian. By selecting the filtering operation to also be of Gaussian form (recall the transform of the Gaussian produces a Gaussian) smoothing of the time properties can be minimized. (A formal discussion of Gaussian smoothing of the WD is provided in Section 2.2.3.)

Figures 3-48 and 3-49 are the respective spDWDs for h(n) a Gaussian, g(n) rectangular, and for both h(n) and g(n) Gaussian. A subtle improvement in Figure 3-49 can be observed. Additional improvement can be effected by increasing M and reducing the variance of g(n), or effectively increasing the variance of the Gaussian filter.

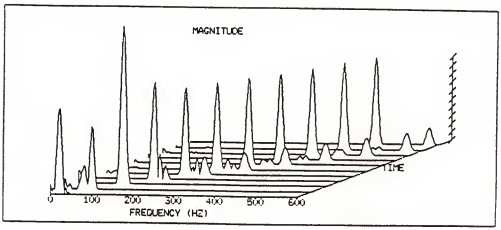

Figure 3-48. Smoothed-psuedo-DWD of multipath chirp example for M=32 and N=256 where Gaussian windowing has been applied.

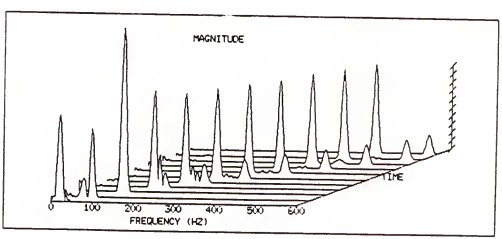

Figure 3-49. Smoothed-psuedo-DWD of multipath chirp example for M=32 and N=256 where Gaussian windowing and filtering have been applied.

3.3.3 Examples of Noise Reduction

Figures 3-50 and 3-51 are the respective STP and DWD of a weak tone in the presence of additive pseudo-random noise for a signal-to-noise ratio of -10db (Figure 3-52). In the STP the noise greatly obscures the tone. In the DWD amplification of the noise precludes observation of the tone. Figure 3-53 is the corresponding spDWD for M=32 (g(n) rectangular) and Figure 3-54 is the spDWD for smoothing over the length of the window.

Figures 3-55 and 3-56 are the STP and DWD of the chirp plus noise in Figure 3-57. Figure 3-58 is the spDWD for both smoothing parameters weighted as a truncated Gaussian where truncation occurs at the variance. By truncating the window at the variance the signal lobe remains narrow; otherwise, the signal and noise resolution would degrade unless a longer tranform length was applied. (Recall, however, the quasi-stationary approximation which limits the transform length for the STP, even for the case of windowing, does not apply to the DWD and a longer transform length may be a viable option.) Alternatively, Gaussian weighting of the filter would require a larger value of M to sufficiently smooth the noise. However, a larger value of M will widen the lobe as the temporal characteristics become smoothed unless the variance is made sufficiently small. In addition, if the value of M is limited by the spDWD processor truncation can be effectively employed.

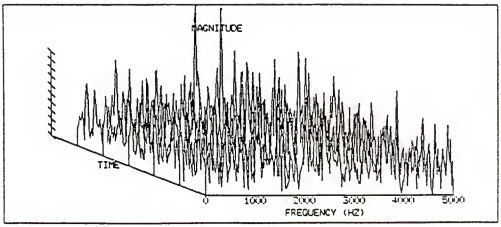

Figure 3-50. STP of sinusoid with additive noise example for N=256.

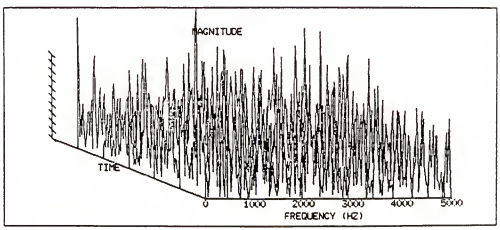

Figure 3-51. DWD of sinusoid with additive noise example for N=256.

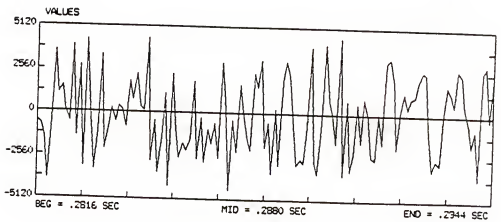

Figure 3-52. Sinusoid with additive psuedo-random noise for a signal-to-noise ratio of -10db.

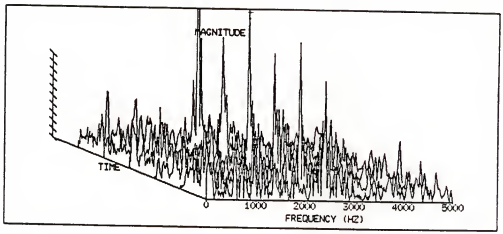

Figure 3-53. Smoothed-psuedo-DWD of sinusoid with additive noise example for M=32 and N=256.

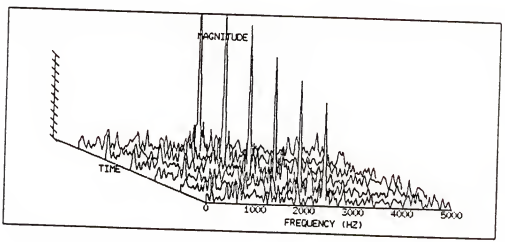

Figure 3-54. Smoothed-psuedo-DWD of sinusoid with additive noise example for M=128 and N=256.

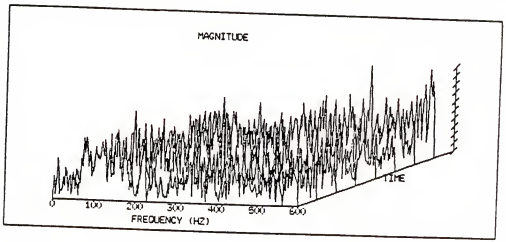

Figure 3-55. STP of chirp plus noise example for N=256.

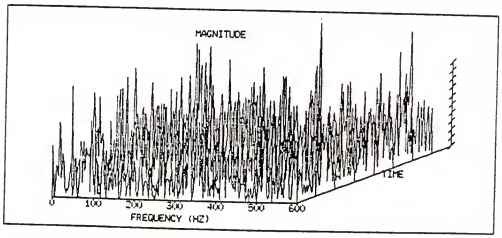

Figure 3-56. DWD of chirp plus noise example for N=256.

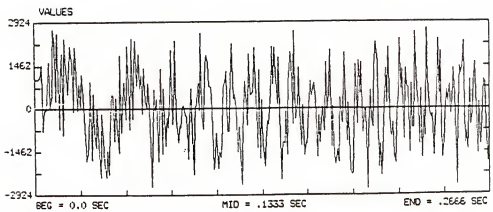

Figure 3-57. Cosine chirp with additive psuedo-random noise for a signal-to-noise ratio of -3db.

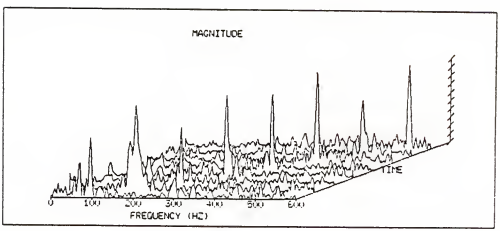

Figure 3-58. Smoothed-psuedo-DWD of chirp plus noise example for M=12 and N=256 where Gaussian windowing and filtering have been applied.

3.3.4 Multipath Radio Signal with Receiver Noise

As a final example, consider the multipath signal of Figure 3-39. In practical applications the radio receiver introduces an additive zero mean Gaussian noise term [37]. Figure 3-59 is the resulting DWD when severe receiver noise (Figure 3-60) is added to the composite chirp plus reflected chirp of Figure 3-39 to produce the corrupted signal in Figure 3-61. Figures 3-62 and 3-63 are the respective spDWD and spDWD with Gaussian smoothing for M=32. In the spDWD the presence of the reflected signal remains obscured; Gaussian smoothing offers some improvement. A large value of M effects too much smoothing of the temporal properties of the signal and serves only to degrade the resultant output (Figures 3-64). However, by applying a large value of M with Gaussian smoothing where the standard deviation (STD) of the Gaussian filter is relatively large or, equivalently, the STD of the temporal weight is relatively small, satisfactory noise reduction can be achieved and the reflection component becomes visible (Figure 3-65). Figure 3-66 is the spDWD for no Gaussian smoothing where the value of M is equal to the STD of the temporal weight in Figure 3-65. Here the reflection component remains obscured and both the direct and reflected signal components are distorted.

The spDWD is sensitive to selection of the filter STD. Reducing the STD of the Gaussian filter slightly (Figure

3-67) reduces the cross-term components which have been amplified by the noise. However, selecting too small a STD for the Gaussian filter will produce too much temporal smoothing on the signal as shown in Figure 3-68. Figures 3-69 and 3-70, the STP for rectangular and Hamming windows, have been added for the sake of comparative completeness. Application of the Hamming window offers some improvement. Still, in both figures the noise precludes observation of the reflection component.

Chapter 4 will concentrate on practical means of implementing the WD, the DWD, and the spDWD for both real and complex signals.

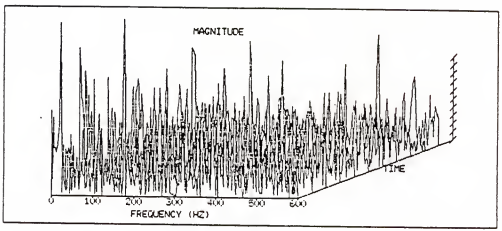

Figure 3-59. DWD of multipath chirp with receiver noise example for N=256.

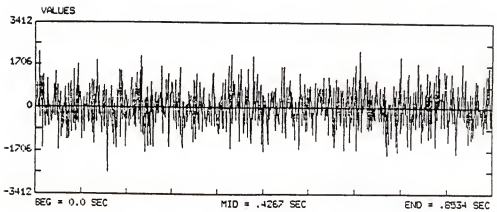

Figure 3-60. Zero mean Gaussian noise with a standard deviation of 700.

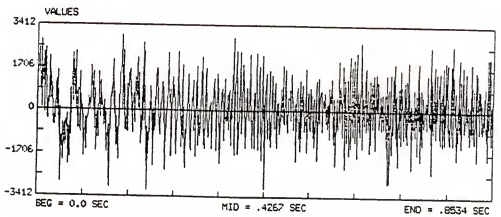

Figure 3-61. Multipath radio with receiver noise example: cosine chirp envelope with attenuated reflection delayed in time and shifted in frequency where τ_d=T/6 and f_d=200Hz in the presence of additive Gaussian noise.

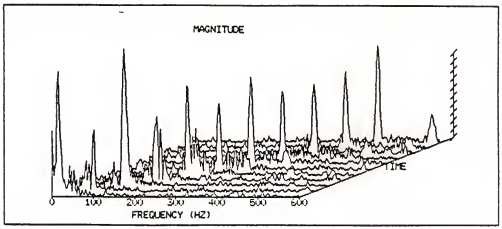

Figure 3-62. Smoothed-psuedo-DWD of multipath chirp with receiver noise example for M=32 and N=256.

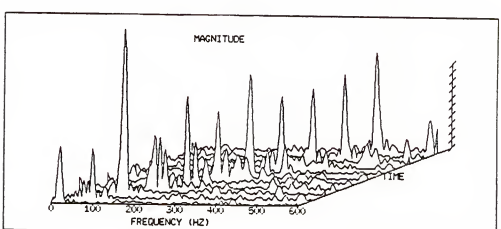

Figure 3-63. Smoothed-psuedo-DWD of multipath chirp with receiver noise example for M=32 and N=256 where Gaussian windowing and filtering have been applied.

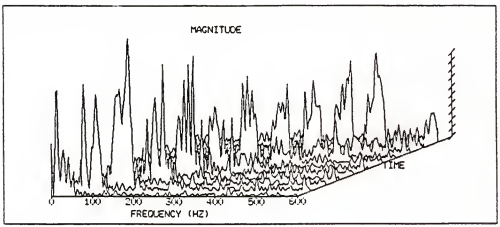

Figure 3-64. Smoothed-psuedo-DWD of multipath chirp with receiver noise example for M=128 and N=256.

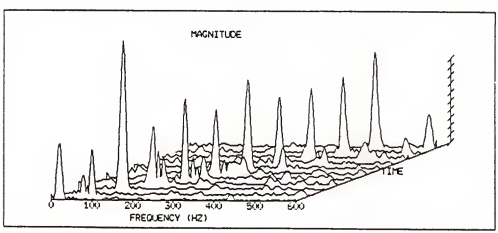

Figure 3-65. Smoothed-psuedo-DWD of multipath chirp with receiver noise example for M=128 and N=256 where Gaussian windowing and filtering have been applied and the STD of the filter is 18Hz.

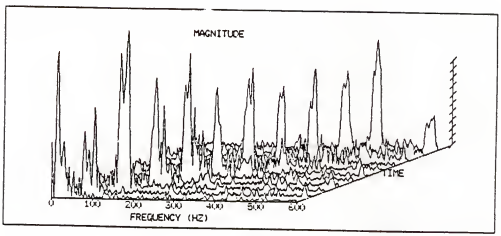

Figure 3-66. Smoothed-psuedo-DWD of multipath chirp with receiver noise example for M=64 and N=256.

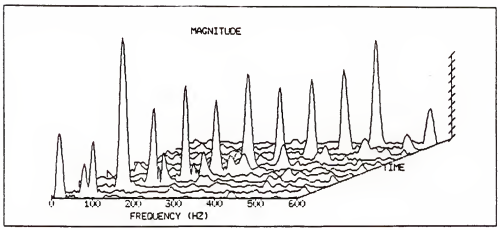

Figure 3-67. Smoothed-psuedo-DWD of multipath chirp with receiver noise example for M=128 and N=256 where Gaussian windowing and filtering have been applied and the STD of the filter has been reduced to 11Hz.

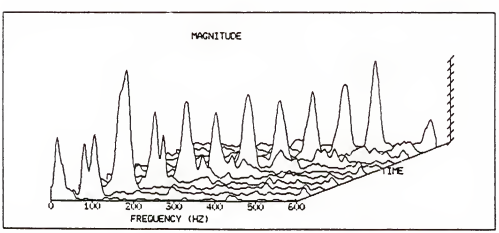

Figure 3-68. Smoothed-psuedo-DWD of multipath chirp with receiver noise example for M=128 and N=256 where Gaussian windowing and filtering have been applied and the STD of the filter has been reduced to 4Hz.

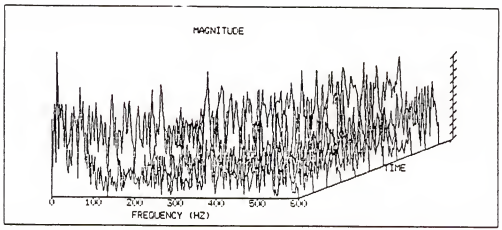

Figure 3-69. STP of multipath chirp with receiver noise example for N=256.

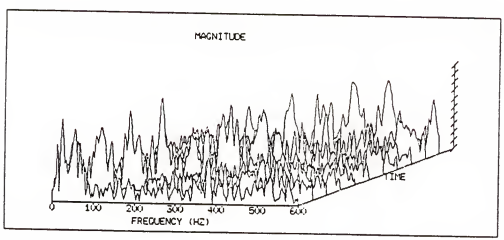

Figure 3-70. STP of multipath chirp with receiver noise example for N=256 where Hamming windowing has been applied.

CHAPTER 4

WIGNER IMPLEMENTATIONS

4.1 Wigner Processing Using Standard FFTs

4.1.1 General Wigner Processing

An attractive feature of the DWD is that it can be computed via the DFT of an inner product function (or for the spDWD a weighted inner product function) defined by the the inner product of the signal shifted in time and its complex conjugate reversed in time and referred to as the DWD kernel. In this form the DWD (and, equivalently the spDWD) readily lends itself to FFT implementations. In short, a DWD processor comprises two major subsystems: kernel generation and a DFT.

Kernel generation for the DWD is relatively straightforward; windowing is performed prior to computation of the inner product. The input sequence is loaded into an N point buffer. Once the buffer is loaded,

one half of the DWD kernel is generated by multiplying the initial data point of the input sequence x(0) with the last point of the input sequence x(N-1) to form the first point of the DWD kernel, multiplying x(1) with x(N-2) to form the second point etc., until the kernel from 0 to N/2-1 is completed. For real data the DWD kernel is symmetric about its center and each product forms two kernel values. For complex data the kernel is congugate symmetric about the center. Durrani et al. [38] presented a systolic architecture for small N DWDs of real data. As discussed in Chapter 2, for real input data rearrangement is not actually needed in the DWD computation.

In Section 3.3 the spDWD was advanced and shown to maintain several attractive features; especially for the two primary conditions under which the DWD produces poor results; multicomponent signals and noisy environments. Despite the advantages of the spDWD, the increased computational requirements associated with producing the spDWD kernel are clear from its definition. Flandrin and Martin proposed a straightforward hardware implementation for the spDWD capapable of "quasi-real-time" data rates [39].

This section will present a high speed concurrent architecture composed of single task processor cells to generate the spDWD kernel. Each cell operates independently; once all inputs (or operands) have been

supplied, the processor task is performed then output as an operand to the other adjacent cell. For illustrative purposes, an N=4, M=3 point processor is given in Figure 4-1. The first set of cells perform complex conjugate multiplication on the input. The output products, i(m,n), from these cells constitute as operands to the complex multiply-add cells. For each multiply-add cell, the operand i(m,n) is shifted in and multiplied by the temporal weight g(m) which has also entered the cell. The product is then added to the composite output, or sum of products elements that have entered the cell from the adjacent multiplier cell to form the new sum of products element, $S_n \leftarrow S_n + g(m)i(m,n)$, that will be input to the other adjacent cell. After an intial startup delay of 2M+1 complex multiply operations, the total smoothed WD inner product output for each consecutive time slice is supplied to the DFT buffers after each operation period. The startup delay is minimal compared to the savings afforded by having continuous output after the initial startup.

In most applications M is typically much smaller than N to avoid masking the nonstationary characteristics of the signal and longer length spDWDs are computed by setting N=N'N'' and completing several passes through the processor to effect the total WD smoothed inner product output. (By removing the weighted terms at m=0 and feeding the composite product into S_n, temporal smoothing may be

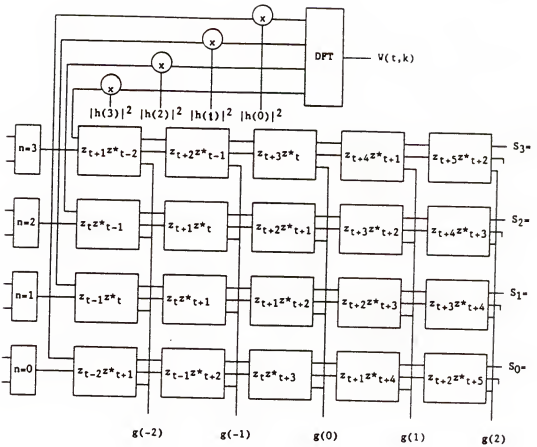

Figure 4-1. N=4, M=3 smoothed-psuedo-DWD processor.

similarily decomposed. Each weighted term at m=0 is then added prior to windowing.) For real input data the processing task comprises all real multiplies and is greatly simplified. As with the DWD kernel, the weighted kernel is symmetric for real data and conjugate symmetric for complex data about its center.

Naturally, the selection of FFT for the DFT stage of of the Wigner processor is both system and application dependent. Reported work on the FFT is voluminous and an extensive reference listing of efficient FFT algorithms is compiled in [40]. Detailed tutorials on radix-r type FFT decompositions can be found in [41,42] and in more recent texts which concentrate on number theoretic transforms [43,44]. A collection of fundamental papers on NTTs is included in [43]. A firm foundation of the modern FFT is provided by Winograd [45-47]. Efficient FFT FORTRAN programs are provided in [48].

Power-of-two algorithms generally require more computations than the number theoretic transforms; however the regularity of the butterfly structure and the ability to perform in place computations has made the radix-2 and radix-4 algorithms a popular choice. Recently the split-radix FFT has become popular because it reduces the number of additions and nontrivial multiplications while maintaining the regularity, flexibility, and ability to perform in place computations associated with the

power-of-two algorithms. The split radix FFT is based on decomposition such that a radix-2 map is applied to the even indexed terms and a radix-4 map is applied to the odd indexed terms. The split-radix is detailed in [49-51] and efficient programs are provided.

4.1.2 Computational Reduction for Real Data

The major computational burden associated with producing either the DWD or spDWD of real data sequences has traditionally been the FFT stage of processing. Chester [36] proposed a DWD analyzer for which the inner product and associated windowing, data rearrangement, etc. was computed via software then applied to an FFT that was constructed using special purpose hardware. In his implementation Chester used the symmetry property of the DWD inner product to compute two DWDs during one FFT cycle. Consequently, Chester concluded that the DWD has nearly twice the throughput of the STFT because two DWDs can be computed via one FFT. A counter to this claim has been that for real data an N point DFT can be computed via an N/2 point FFT because the corresponding output is conjugate symmetric [41]. Using the symmetry of both the DWD inner product input and the corresponding DWD output, two DWDs can, in fact, be computed via one N/2 point FFT.

To see this let $i_1(m)$ and $i_2(m)$ each denote respective DWD kernels. At the output stage of the kernel generator

each kernel sequence is redefined as an N/2 point sequence as follows:

$$r_n(m)=r_{ne}(m)+r_{no}(m);m=0,1,\ldots,N/2-1;m=1,2, \quad (4\text{-}1a)$$

where

$$r_{ne}(m)=i_n(2m) \ ;n=1,2 \quad (4\text{-}1b)$$

are even funtions and

$$r_{no}(m)=i_n(2m+1)-i_n(2m-1);n=1,2 \quad (4\text{-}1c)$$

are odd functions. The DFT of $r_n(m)$ n=1,2 expresses as follows:

$$R_n(k)=R_{ne}(k)+R_{no}(k);n=1,2 \quad (4\text{-}2)$$

where $R_{ne}(k)$ and $R_{no}(k)$ are the respective DFTs of $r_{ne}(m)$ and $r_{no}(m)$ for n=1,2. The first term $R_{ne}(k)$ is equivalent to the DFT over m even of the original sequence $i_n(m)$ as below:

$$R_{ne}(k)=2/N \sum_{m=0}^{N/2-1} i_n(2m)e^{-j4\pi mk/N}$$

$$=2/N \sum_{\substack{m=0 \\ m \text{ even}}}^{N-1} i_n(m)e^{-j2\pi mk/N};n=1,2. \qquad (4-3)$$

The second term $R_{no}(k)$ reduces as follows:

$$R_{no}(k)=2/N \sum_{m=0}^{N/2-1} [i_n(2m+1)-i_n(2m-1)]e^{-j4\pi mk/N}$$

$$=2/N\sum_{\substack{m=0 \\ m \text{ odd}}}^{N-1} i_n(m)e^{-j2\pi(m-1)k/N}$$

$$-2/N[\sum_{\substack{m=0 \\ m \text{ odd}}}^{N-1} i_n(m)e^{-j2\pi(m+1)k/N};n=1,2.$$

$$=2/N(e^{-j2\pi mk/N}-e^{j2\pi mk/N})\sum_{\substack{m=0 \\ m \text{ odd}}}^{N-1} i_n(m)e^{-j2\pi mk/N};m=1,2. \qquad (4-4)$$

Now, by applying $r_1(m)$ to the real input of the N/2 point FFT and $r_2(m)$ to the imaginary part of the N/2 point FFT, the effective input is the following:

$$y(n)=r_1(m)+jr_2(m);m=0,1,...,N/2-1 \qquad (4-5)$$

with corresponding output,

$$Y(k)=R_{1e}(k)+jR_{1o}(k)+jR_{2e}(k)+R_{2o}(k);$$

$$k=0,1,...,N/2-1, \qquad (4-6)$$

by definition of the DFT. Applying the folding properties at the output of the N/2 point FFT (4-6), the DWDs W(n,k) n=1,2 of the original kernel sequences $i_n(m)$ n=1,2 can be obtained from the sum of the DFT over m even and the DFT over m odd which via Equations (4-3) and (4-4) equate to the following:

$$W(1,0)=1/4Re\{Y(0)+Y(N)\}+1/4Im\{Y(0)-Y(N)\},$$

$$W(2,0)=1/4Im\{Y(0)+Y(N)\}+1/4Re\{-Y(0)+Y(N)\}; \qquad (4-7a)$$

for k=1,2,...,N/4,

$$W(1,k)=1/4\,Re\{Y(k)+Y(N-k)\}+1/4\,Im\frac{\{Y(k)-Y(N-k)\}}{2\sin(2\pi k/N)},$$

$$W(2,k)=1/4\,Im\{Y(k)+Y(N-k)\}+1/4\,Re\frac{\{-Y(k)+Y(N-k)\}}{2\sin(2\pi k/N)}; \qquad (4\text{-}7b)$$

for k=N/4+1,...,N/2-1,

$$W(1,k)=1/2\,Re\{Y(N/2-k)\},$$

$$W(2,k)=1/2\,Im\{Y(N/2-k)\}; \qquad (4\text{-}7c)$$

and,

$$W(1,N/2)=1/4\,Re\{Y(0)+Y(N)\}+1/4\,Im\{Y(0)-Y(N)\}, \qquad (4\text{-}7d)$$

$$W(2,N/2)=1/4\,Im\{Y(0)+Y(N)\}+1/4\,Im\{-Y(0)+Y(N)\};$$

where for real input the DWD is symmetric about N/2.

4.2 High-speed Processing Via Algebraic Mappings

4.2.1 Problem Statement

For many applications restriction to real signals is not practical. In communication systems, for example, demodulation techniques at the receiver produce complex encoded data. In addition, application of an analytic signal can be shown to have certain advantages specifically for DWD processing, several of which are listed below:

elimination of information redundancy associated with real sequences;

elimination of cross terms that result from interference between positive and negative frequencies and degrade dc information;

the effective bandwidth-time product is reduced;

sampling at the Nyquist rate is sufficient to prevent frequency aliasing of the DWD (for real signals sampling must be at least double the Nyquist rate); and,

instantaineous frequency can be obtained from the DWD.

Whether real or complex data are to be used, computation of the DWD is explicitly complex multiply intensive and for the DWD to be a viable instrument in real time applications

requires a processor capable of performing complex operations at very high data rates.

In general, to complete a complex multiply requires two real multiplies and two adds and traditional methods of performing complex arithmetic operations are both hardware complex and of long latency. To overcome this problem alternative number systems have been explored. The residue number system (RNS), due to its potential for high-speed computations, has received much attention. In the residue number system each integer is represented via a set of smaller wordlength integers, called residues.

The attractive feature of the RNS is that multiplication (and/or addition) can be completed by independently performing multiplication (and/or addition) on each of the smaller wordlength residues. Therefore, in the RNS each operation can be performed via table-lookup using high-speed semiconductor memory. Numerous papers have been published on application of the RNS to transform design [52-57]. In most cases, however, the RNS has not achieved a distinct advantage over conventional architectures. The principle obstacle to achieving an advantage has been a high scaling and residue to decimal conversion overhead, whether using multi-moduli Chinese remainder theorem (CRT) or mixed-radix conversion (MRC) routines. That is, the RNS is an integer system; unless some form of magnitude scaling is performed at intermediate

stages of the processing task the dynamic range extension requirements become inordinate. Unfortunately, magnitude scaling requires conversion from RNS which is a cumbersome operation. This fact has limited the successful application of the RNS in DSP.

If the moduli used to produce each residue satisfy certain constraints a modular (residue) system called the quadratic residue numbering system (QRNS) is defined. In the QRNS an isomorphism exists for which the number of real multiplies and adds necessary to perform a complex multiply can be reduced by two and one respectively. For this reason, despite the scaling overhead associated with RNS based implementations, application of the QRNS to complex multiply operations is warrented. Recently Taylor [58] introduced a single-modulus complex arithmetic logic unit that utilized the QRNS. Taylor's SM-QRNS reduced the previous obstacle of multi-moduli RNS, namely residue-to-decimal conversion, down to a set of trivial shift-adds [58]. As a result, both fast and hardware elegant computational SM-QRNS primitives could be designed.

A smoothed-pseudo-DWD processor implementation was shown to be well suited to the SM-QRNS [59]. The DWD was shown to maintain certain properties that make it especially attractive to the QRNS and a reduction in computation time for the SM-QRNS was mathematically conjectured. The question of nested multiplies in the QRNS

environment and associated magnitude scaling overhead was advanced and alernative FFT structures to reduce nested multiplies were proposed for the DFT stage of the DWD processor.

In this section, a high performance SM-QRNS DWD estimator that uses standard FFT structures will be shown to be a practical reality. DWD estimators for both radix-2 and radix-4 DIT FFT subsystems will be provided and quantified in terms of increase in total system throughput relative to transform length and wordsize. The SM-QRNS will be shown to reduce the effective time delay of the spDWD kernel generator proposed in Section 4.1 for the case of complex data. Because the temporal smoothing parameter is optimally real-symmetric the SM-QRNS kernel generation in [59] will be modified to reduce the hardware for longer wordlengths. First, the principles of the QRNS and its corresponding isomorphism for multiply reduction will be developed.

4.2.2 Quadratic Residue Number System Principles

Given a finite set of integers, Z_M, each integer $a \varepsilon Z_M$ has a unique RNS representation given by:

$$a \rightarrow (a_1, a_2, \ldots, a_L);\ a_k = a \bmod p_k \varepsilon S_k$$

where

$$M=\prod_{k=1}^{L} p_k^{e_k};\ \gcd(p_i,p_j)=1. \qquad (4\text{-}8)$$

Signed numbers, $a\varepsilon[-M/2,M/2)$, are represented as in (4-8) if $a\geq 0$; if $a<0$, then $a_k=(M-|a|)\bmod p_k$. The residue sets $(Z_k,+,\cdot,0,1)$, $k=1,2,\ldots,L$ each form a field and integer addition and multiplication can be computed via L independent modulo p_k operations. A brief overview of the Residue Numbering System (RNS) is provided in Appendix A.

The set of complex numbers $\{a+b\sqrt{-1};\ a,b\varepsilon Z\}$ form a subring generated by Z and $i=\sqrt{-1}$ and are called the ring of Gaussian integers. In the RNS, the corresponding sets of residues modulo p_k are defined as $\{a_k+b_k i_k;\ a_k,b_k\varepsilon Z_{p_k}\}$, $k=1,2,\ldots,L$ and each forms a field $(Z_{p_k}[i_k],+,\cdot,0,1)$ under multiplication and addition. If $i_k^2\equiv -1 \bmod p_k$ has no solution then $i_k\notin Z_{p_k}$ and -1 is called a quadratic non-residue modulo p_k. A numbering system for which $i_k\notin Z_{p_k}$ for all k is referred to as the complex-RNS (CRNS). In the CRNS each $a_k+b_k i_k$, $k=1,2,\ldots,L$, is considered to comprise real and imaginary parts where each product requires four real modulo p_k multiplies and two real modulo p_k adds. Conversly, if $i_k^2\equiv -1 \bmod p_k$ has a solution, then $i_k\varepsilon Z_{pk}$ and is a quadratic root, -1 is a quadratic residue. A numbering for which $i_k\varepsilon Z_{pk}$ for all k is referred to as the quadratic RNS, or QRNS, and the residue values $a_k+i_k b_k$ are treated as real.

If i_k is a quadratic root then there exists an isomorphism $\alpha: z_{pk}[i_k] \rightarrow z_{pk}^2$ where $z_{pk}^2 = z_{pk} \times z_{pk}$ is the product set of z_{pk} with itself. Each product forms a corresponding set of two-tuples $z_{pk}^2 = \{(a_k, b_k);\ a_k, b_k \varepsilon z_{pk})\}$, where the rules of composition satisfy

$$(a_k, b_k) \phi (c_k, d_k) = ((a_k \phi c_k), (b_k \phi d_k));\ k = 1, 2, \ldots, L, \tag{4-9}$$

for $\phi = (+, -, *)$, each taken modulo p_k. Here addition also requires two real adds; however, multiplication requires only <u>two</u> real multiplies and <u>no</u> real additions. The isomorphism $\alpha: z_{pk} \rightarrow z_{pk}^2$ and its application to high-speed concurrent architectures for DSP have been covered extensively in the literature [55-57,60]

If p_k is a Gaussian prime $p_k = 4K+1$, or a composite of Gaussian primes, the isomorphism is the following,

$$\alpha((a_k + b_k j_k)) = ((a_k + b_k j_k) \bmod p_k, (a_k - b_k j_k) \bmod p_k) \dot{\mp} (z_k, z_k^*);$$

$$a_k = (2^{-1}(z_k + z_k^*)) \bmod p_k;\ b_k = 2^{-1} j^{-1} (z_k - z_k^*)) \bmod p_k;$$

$$j_k^2 = -1 \bmod p_k;\ k = 1, 2, \ldots, L, \tag{4-10}$$

where $j_k \varepsilon z_{pk}$ is a quadratic residue. Although this is a very recent innovation, a number of multi-moduli papers ($L \geq 2$) have appeared in print. The significant advantage to the QRNS is the computational reduction allowed by (4-10) and the term QRNS as used in the engineering literature implies application of the isomorphism in (4-10).

4.3.3 Single Modulus System

Recently, Taylor [58] developed the QRNS for a single modulus (L=1) to form what is called the SM-QRNS. In the SM system Taylor was able to overcome the principle performance limitation to most RNS systems, that of residue to decimal conversion. The moduli used to define the SM-QRNS were primes of the form $p=2^n+1$ where admissable values of n are 2, 4, 8, 16, or 32. In this form the SM-QRNS parameters are highly composite and essentially radix-2 in value (i.e. require a simple correction stage).

For $p=2^n+1$ the coefficients in the isomorphism (4-10) equate to $2^{-1}=2^{n-1}+1$; $j=2^{n/2}$; and $(2j)^{-1}=2^n+1-2^{n/2}$. Each of these SM-QRNS scaling modules can be implemented as binary shift and/or operations as below:

scaling by j

$$jX \bmod p = 2^{n/2}X_{LO} - X_{HI};\ X = 2^{n/2}X_{HI} + X_{LO}$$

scaling by 2^{-1}

$$2^{-1}X \bmod p = (2^{n-1}+1)X_{LO} + X_{HI};\ X = 2X_{HI} + X_{LO}$$

scaling by $(2j)^{-1}$

$$(2j)^{-1}\text{modp}=X_{HI}-2^{(n/2-1)}X_{LO};\ X=2^{(n/2+1)}X_{HI}+X_{LO}. \quad (4\text{-}11)$$

In order to compete with traditional complex arithmetic units, the computational units that make up the SM-QRNS DWD processor must be designed to operate with high speed and low hardware complexity. To explore this question, a detailed analysis of each QRNS single modulus subsystem architecture was performed in [58]. Based on an elementary delay of a primitive gate (e.g. NAND TTL gate), these architectures were reported to integrate into an efficient arithmetic unit with a substantially reduced hardware latency budget. To test this hypothesis a gate array SM-QRNS unit, based on GE's IGC20000 technology was undertaken [61]. The timing studies in [61] confirmed the conjectures advanced in [58]. Based on these computational primatives, a high resolution complex intensive multiply DWD estimator can be designed to achieve high throughputs. Furthermore, the symmetry property of the DWD kernel and the corresponding real-valued nature of the DWD output will be shown to integrate well with the SM-QRNS.

4.2.4 Wigner Processor Implementation

Figure 4-2 illustrates the SM-QRNS DWD processor. The input data to the SM-QRNS encoder are assumed to be n-bit 2's complement words. Each number is mapped into an (n+1)-bit integer modulo p. If the input value X<0, then $-X=TC(X)+2^nX$ (TC denotes a 2's (radix) complement operation) having an n-bit format. The same integer in the SM-QRNS case is given by $-X\equiv(2^n+1-X) \bmod p=TC(X)+1$. For example, if p=17 and X=-1, then TC(1)→1111 and $TC(1)+1=16\equiv-1 \bmod 17$. If N=-2, then TC(2)→1110 and $TC(2)+1=1111=15\equiv-2 \bmod 17$, etc. Therefore, the code converter can be implemented by modifying a conventional 2's complement encoder with a simple increment by 1 circuit.

Data windowing is performed prior to two-tuple QRNS conversion. The complex windowed data points $x_n(m)$ and $x_n(-m)$ are mapped into their respective SM-QRNS two-tuple equivalents via the isomosphism in (4-10) for L=1. Computation of the first two-tuple element, z, requires two modular operations: j-magnitude scaling and modulo p addition. The first operational component, the j scalar, performs scaling on b by $2^{n/2} \bmod (2^n+1)$ which is implemented as in (4-11). The modulo p adder comprises three parts: an n-bit fast carry-lookahead adder, a modulo p mapping

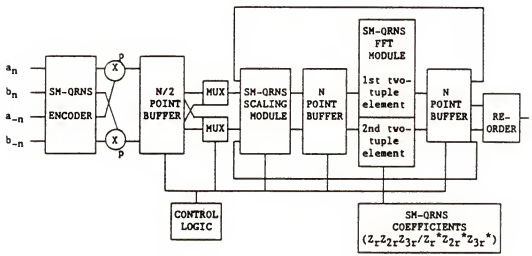

Figure 4-2. SM-QRNS DWD processor block diagram.

unit, and control logic (compare network) [58]. The n-bit adder is a conventional full carry-lookahead adder. The compare network determines whether modulo p mapping is necessary. If $S \geq p$, then the modulo p mapping unit performs the map $S \leftarrow S-p$ using the procedure detailed in reference [52]:

The second two-tuple element, z^*, requires inclusion of a negator prior to modulo p addition, where the negator completes the following operation: $x \rightarrow -x=(p-|x|) \bmod p$.

The only modular components necessary to generate the DWD kernel are modulo p multipliers. The modulo p multiplier is realized via an unsigned multiplier and a network for special case processing [58]. For inputs x and y the special cases are the following:

if both x and y equal 2^n, then set their product to 1;

if x equals 2^n but y does not, then set their product to -y,

where the negator is used to obtain -y. If the single modulus is sufficiently small ($p=2^n+1$; $n \leq 8$), table-lookup methods may be employed where a 2(n+1)-bit address is compressed into a 2n-bit address by using logic gates to discern the case 2^n from 0. In [58] Taylor describes efficient direct lookup methods to extend the allowable moduli to satisfy $n \leq 16$.

Complex conjugate multiplication is completed in the

SM-QRNS by multiplying, modulo p, the first two-tuple element of each data point to the second two-tuple element of the other data point. Each operation of this form produces effectively two complex kernel output values. That is, by definition the DWD is conjugate symmetric, which in the SM-QRNS translates to $(z_i(n),z_i^*(n))=(z_i^*(-n),z_i(-n))$, therefore, each QRNS multiply provides two kernel output values where the kernel values are related by the transposition of the two-tuple elements. The appropriate DWD kernel output is magnitude scaled then supplied to the SM-QRNS FFT modules for each two-tuple element. After the first complete pass of data through the FFT modules DWD kernel generation for the next consecutive window center position is initiated.

The magnitude scaling module (Figure 4-3) comprises three stages: conversion from the two-tuple values; scaling; then conversion back into the two-tuple values. Magnitude scaling is performed prior to each butterfly stage. Effectively then, magnitude scaling is associated with each multiply in order to control the geometric dynamic range growth of the RNS products (analogous to full precision products in conventional systems). For example, let the output of the two complex butterfly inputs Z_1 and Z_2, where $Z_j=X_j+iY_j$, be Z=A+iB and let Z be encoded as a <u>signed</u> SM-QRNS word with respect to a modulus $p=2^n+1$ (prime) such that $(p-1)/2=2^{n-1}$. Then set

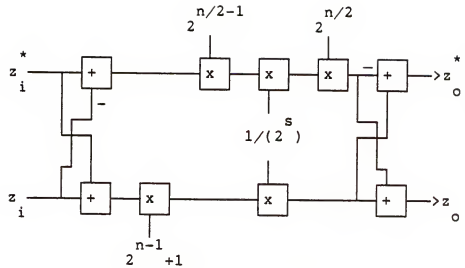

Figure 4-3. Magnitude scaling module.

$\max(|X_j|)=\max(|Y_j|)=V$, where $\max(|A|)=\max(|B|)=2V^2+N_s 2V \leq 2^{n-1}$ for N_s the number of additive stages in the butterfly module. At the initial pass, multiplication is by one and magnitude scaling (associated with the additive operations) is incorporated into the second level.

The SM-QRNS FFT modules for each two-tuple element are independent and identical by definition of multiplication and addition in (4-9). Relative to a conventional radix-4 FFT design, for example, (Figure 4-4), the QRNS/FFT butterfly reduces from a four stage process (made up of 12 real multiplies, followed by three stages of add/subtracts) to two independent three stage processes (each of 3 real multiplies followed by four add/subtracts (+ one -j scaling operation) then four add/subtracts)(Figure 4-5). The SM-QRNS design relies on nearly modulo 2^n hardware for all derived mathematical operations [54].

Recall, the DWD is a real-valued function; therefore, no conversion from SM-QRNS is needed at the output. This fact can be coupled with the independent and identical relation between the SM-QRNS FFT modules for each two-tuple element to increase the throughput in the DWD processor.

For clarity, consider a power-of-2 FFT. The butterfly equations for x complex express as

$$x^{s+1}(i) = x^s(i)+W^r x^s(j)$$

$$x^{s+1}(j) = x^s(j)-W^r x^s(i) \qquad (4\text{-}12)$$

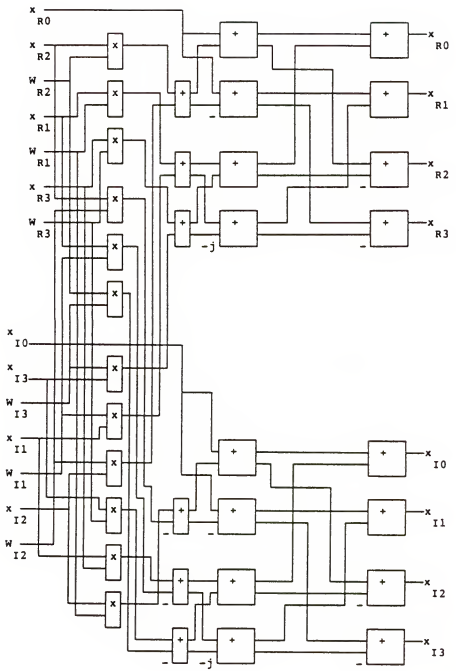

Figure 4-4. Four point FFT module for mod2^n based processor.

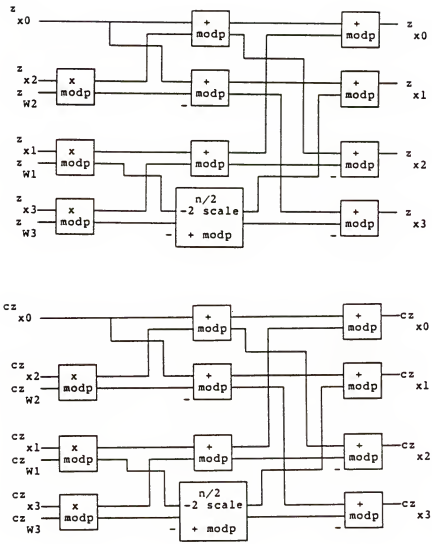

Figure 4-5. Four point FFT module for SM-QRNS based processor.

where the superscripts denote the butterfly stage and W^r the appropriate twiddle factor. In an SM-QRNS two-tuple equivalent (z,cz) the corresponding butterfly equations are the following

$$z_x^{s+1}(i) = z_x^s(i)+z_W^r z_x^s(j)$$

$$cz_x^{s+1}(i) = cz_x^s(i)-cz_W^r cz_x^s(j)$$

$$z_x^{s+1}(j) = z_x^s(j)+z_W^r z_x^s(i)$$

$$cz_x^{s+1}(j) = cz_x^s(j)-cz_W^r cz_x^s(i) \qquad (4\text{-}13)$$

where the subscripts denote the correspondence of the QRNS value to its integer equivalent (ie x -> (z_x, cz_x)) and the superscripts are as before. Notice, only the first two-tuple element z is needed at the input stage to each butterfly to produce the output z.

For real output the first element of the two-tuple in a SM system is equivalent to the integer value; therefore, only the first two-tuple element needs to be computed (naturally z=cz) and because the DFT operations for each two-tuple element are independent, only those associated with the first two-tuple element are needed in the DWD processor implementation provided DFT processing is performed entirely in the SM-QRNS. That is, for the single modulus system magnitude scaling can be performed via

table-lookups applied directly to the first two-tuple values, eliminating the need for conversion from the SM-QRNS two-tuple elements during intermediate processing. Here the SM-QRNS butterfly module for the second two-tuple element is employed to speed up processing by applying first two-tuple elements and supplying their appropriate SM-QRNS twiddle factors.

Alternatively, two DWDs can be simultaneously computed using one SM-QRNS DFT by composing alternating two-tuple elements for even and odd values of t, where t denotes window position, such that each two-tuple applied to the DFT is of the form (z_t, cz_{t+1}) $t=0,2,4,\ldots,N-2$. The first element of the corresponding two-tuples at the output of the DFT are the DWD values at t and the second two-tuple elements are the DWD values at t+1.

For longer wordlengths ($n \geq 16$) where table-lookup scaling operations are not feasible, conversion from the two-tuples may be necessary. Here both two-tuple elements must be maintained throughout the DFT processor until the final stage of processing during which time only the first two-tuple element is needed. At the final pass of data, the SM-QRNS components for the second two-tuple elements are employed to speed up processing.

As the DWD avoids the traditional STFT trade-off problem between window length and the assumption of stationarity, longer observation times may be chosen for

DWD processing. An essential part of any RNS based system is magnitude scaling and is important to the choice of FFT. For longer transform lengths a large number of nested multiplies, associated with the radix-2 FFT and radix-4 structures, translates to extensive magnitude scaling requirements, which should be avoided in an RNS system. Instead, architectures with fewer nested multiplies should be considered. Taylor and Huang investigated the comparison/trade-off characteristics associated with various standard transform alternatives [53]. In their work, the WFTA proved favorable in terms of reducing nested scaling operations; however, the irregular data flow and extension requirements are a definite disadvantage.

A hybrid system which uses a large radix Cooley-Tukey decomposition with a small-WFTA at each level can be used to reduce nested multiplies while avoiding the major disadvantagees associated with the large-WFTA. For example, a large N point DWD processor using a radix-L CT decomposition would require only $\log_L N$ nested levels of complex multiplication associated with the twiddle factors. Each of these will be computed in the SM-QRNS. If the radix-L DFT is computed via an L point small-WFTA one additional multiply stage, consisting of L multiplies, is added to each level. For the SM-QRNS, the j-scale operations are absorbed into the multiply stage of the small-WFTA and pre-weave and post-weave operations are exclusively addition and subtraction.

4.3.5 Throughput Comparison

Assuming all worst case conditions, for the SM-QRNS (Table 4-1) a direct comparison was made between the SM-QRNS based DWD processor and a conventional DWD processor equivalent. To establish a common denominator, all operations were treated as consecutive and both processors were assumed to have input data in a 2's complement word representation and fixed integer scaling. Tables 4-2 and 4-3 provide respectively the throughput increase for the SM-QRNS DWD processor for the case of a radix-2 SM-QRNS FFT module and a radix-4 SM-QRNS FFT module. Percentage throughput is defined as

$$\frac{(DWD/\Delta)_{SM\text{-}QRNS}-(DWD/\Delta)_{mod2^n}}{(DWD/\Delta)mod2^n}X100 \qquad (4\text{-}14)$$

where Δ is unit of delay. The wordlength for which the SM-QRNS becomes faster is dependent on the transform length and the FFT butterfly size and is $n \geq 5$ or less. For such small word sizes, however, table-lookups may be used to perform magnitude scaling and only the first two-tuple

Table 4-1. Analysis of SM-QRNS DWD processor throughput.

Assumptions in Comparison
(1) comparison is with mod2^n DWD processor
(2) magnitude scaling is required for each multiply
(3) initial conversion and conversion from/to SM-QRNS for each magnitude scaling operation (i.e. no table-lookups
(4) both two-tuple elements must be maintained throughout the processor, until the final pass of data (this follows from (3))
(5) all multiplies are negative (negative multiplies have greatest delay in SM-QRNS)
(6) the symmetry property of the DWD kernel is applied to BOTH processors to reduce computations
(7) in the mod2^n processor, the delay associated with complex conjugation is equivalent to that of negation
(8) operations are treated as consecutive

Table 4-2. The percentage increase in throughput for the SM-QRNS given a radix-2 FFT as the bases for the DFT stage in both processors.

TRANSFORM LENGTH = N	WORDLENGTH = n	
N	n = 16	n = 32
64	54.6	78.6
256	50.2	74.4
1024	48.4	71.9
4096	46.9	70.3
wordlength at which the processing speeds become equivalent: n < 5		

Table 4-3. The percenatge increase in throughput for the SM-QRNS given a radix-4 FFT as the bases for the DFT stage in both processors.

TRANSFORM LENGTH = N	WORDLENGTH = n	
N	n = 16	n = 32
64	71.9	97.1
256	62.9	87.5
1024	57.8	82.0
4096	54.5	78.5

wordlength at which the processing speeds become equivalent:

N	n
64	<3
256	<4
1024	<4
4096	<5

element is needed, translating to a factor of two in throughput increase for the SM-QRNS.

To achieve very high data rates the FFT stage of the processor may be implemented using the butterfly pipeline algorithms developed in [62] and [63] for respective radix-2 and radix-4 FFT implementations. A variable-length delay commutator design approach which overcomes most of the complexity problems associated the pipeline algorithms was provided in [64] and used to develop a delay commutator circuit for a radix-4 FFT pipeline processor. For the radix-4 QRNS/FFT processor, SM-QRNS two-tuple elements for each set of 4 DWD kernel values are pipelined through their respective butterfly modules, a SM-QRNS scaling module, and two SM-QRNS real word delay modules.

4.2.6 Weighted Wigner Kernel Generation

The spDWD complex kernel is explicitly complex computation intensive. The high speed concurrent architecture of Figure (4-2) is used to generate the spDWD kernel. As with the DWD, only one half the spDWD kernel is computed. The values for the other half of the spDWD kernel are determined by transposing two-tuple elements at the input to the SM-QRNS FFT.

SM-QRNS architectures for each of the modular components that make up the SM-QRNS spDWD kernel processor have been provided by Taylor[58] for application to a SM

Complex ALU. Theoretically produced performance measures were conjectured in [58] then experimentally confirmed in [61]. Based on these operation times in a conventional radix-2 system the complex multiplier cells would complete a complex multiplication on the order of (28n-4)Tg, for Tg a unit gate delay, and the multiply-add cells would perform at (14n+6)Tg. In the SM-QRNS the time to complete each complex conjugate multiply will be, under worst case conditions, (14n+34)Tg. If the time to complete conversion from SM-QRNS is included at the complex conjugate multiplier cells the total processing time at that stage will be on the order of, again under worst case conditions, (14n+85)Tg. Assuming the time to effect complex conjugation is essentially the same as negation, the total processing time at the first stage of the radix-2 processor is (28n+1)Tg.

The cells do not operate until all operands have been supplied; therefore, processing speed is primarily dependent on the speed of the slowest cell. In the radix-2 system the first stage of the smoothed kernel processor will greatly impede system throughput, almost by a factor of one-half, whereas inclusion of the SM-QRNS cells produces like processing speeds for large n, minimizing the amount of time cells are standing idle. For smaller wordlengths, kernel generation is performed entirely in the SM-QRNS where magnitude scaling is incorporated directly

into the table-lookup operations. The first two-tuple element for the entire kernel is carried through the kernel generation process, afterwhich the second two-tuple element is obtained, if necessary, via the relation $z_i^*(n)=z_i(-n)$.

If speed is a priority, table-lookups can used to perform the modulo p multiplication. Here, magnitude scaling can be incorporated into the table-lookups. If the number of window selections is reasonably limited (ie rectangular, Hamming, Blackman, Gaussian), table-lookups in the form of scaling operations can be used in place of multiplications in the multiply-add cells, effectively reducing the table size by $2^{n-n'}$ where $2^{n'}$ is the number of available window selections.

4.3 High-frequency, High-resolution Wigner Processing

As with the continuous-time case, filtering and modulation are important DSP operations. Fortunately, if the respective spectrums of both x(n) and h(n) are assumed to vanish over one-half the interval, either by oversampling or in a case as analytic signals, the properties of filtering and modulation are analogous to their continuous-time counterparts. If $W_y(t,k)$, $W_x(t,k)$, and $W_h(t,k)$ are the respective DWDs of y(n), x(n), and h(n), then

$$y(n)=x(n)*h(n)$$

$$\Rightarrow W_y(n,k)=1/N\left[\sum_{m=0}^{N-1} W_x(n-m,k)W_h(m,k)\right]R_N(m) \qquad (4\text{-}15)$$

and,

$$y(n)=x(n)h(n)$$

$$\Rightarrow W_y(n,k)=1/N\left[\sum_{i=0}^{N-1} W_x(n,k-i)W_h(n,i)\right]R_N(i)$$

where

$$R_N(n)=\begin{cases}1, & 0\leq n\geq N-1\\ 0, & \text{otherwise.}\end{cases} \qquad (4\text{-}16)$$

The filtering and modulation properties can be used to produce a band-select Wigner spectrum. That is, like the FFT the DWD is a baseband analysis tool. Many processing applications, however, involve concentration on a selected band of frequencies away from baseband. To achieve sufficient spectral resolution in such cases may require more sample points than can be handled efficiently by the

DWD analyzer. However, because an FFT is imbedded in the DWD stucture, a band-selectable (or pseudo-zoom) DWD can be designed to produce fine resolution spectral analysis over a band of prespecified frequencies [65].

Figure 4-6 illustrates the pseudo-zoom process. Prior to computation of the Wigner kernel, the data is bandpass filtered over the frequency band of interest; this so called "observation band" is heterodyned to baseband, lowpass filtered then decimated in time. The DWD of the modulated output is then computed.

The bandpass filter passes only the observation band and prevents overlap of the signal into the observation band during the heterodyning process. The DWD is a real-valued function which does not preserve phase information of the signal. That is, recovery of the signal from the DWD can be obtained only up to a constant. For this reason if the signal type to which the process is being applied is unlikely to undergo a 180 degree change in phase over the observation band relaxed phase conditions may be warrented to improve the passband flatness, and rolloff and attenuation characteristics. Otherwise, linear phase filters are needed.

The term "pseudo" is used because by filtering out all but the frequency band of interest some cross-terms resulting from the inner product that would normally fall within the observation band have been eliminated. Recall

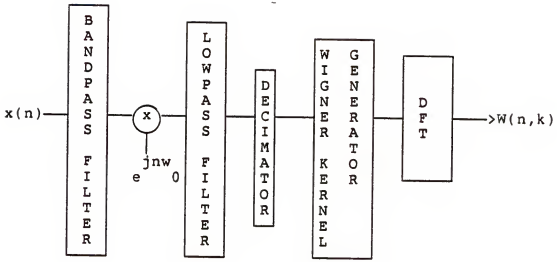

Figure 4-6. Block diagram of band-select DWD process.

(Equation (3-22)), when the Wigner kernel is formed all of the sinusoidal components are mapped into the form $1/2(\cos 2\omega \pm 1)$ where ω is the original frequency of the sinusoid, or into the form $\cos(\omega_1 \pm \omega_2)t$, where ω_1 and ω_2 are the frequencies of two component sinusoids with two different frequencies. If ω_1 is within the observation band, however ω_2 is not, the cross term contributions at $\omega_1 \pm \omega_2$ do not appear in the DWD of the band-select signal.

For real signals, frequencies are mapped to dc during computation of the Wigner kernel, effectively corrupting the signal around dc, and dc information is difficult to obtain from the DWD (Section 3.3). Therefore, insertion of a "guard band" is needed during the heterodyning process; that is, the observation band is modulated to "near" baseband, where the guard band is system and application dependent [65]. For example, consider the case of rectangular windowing with acceptable spectral degradation -26db: for the rectangular window an expected rollof rate is 13db per sidelobe and the guard band must be greater than the width of the second sidelobe. For analytic signals, the guard band is not needed (Section 3.3).

The lowpass filter passes only frequencies within the analysis band plus the guard band and filters out any unwanted frequencies outside the band of interest that result from folding during the heterodyning process. As with the bandpass filter, the filter characteristics are system and application dependent.

Finally, the decimator is used to reduce the effective sampling rate of the input to the DWD processor to achieve increased resolution. The decimator type is also system and application dependent and is primarily a trade-off between resolution efficiency and system complexity.

After the band-select process is complete, the preprocessed output is input to the kernel generator of the DWD.

To illustrate the "pseudo"-zoom process, a music signal sampled at 20kHz (Figure 4-7) will be observed over a 1kHz band between 5500 and 6500Hz. The filter characteristics used in the band-select DWD example are given in Figures 4-8 and 4-9. Decimation is effected as a simple switch and performs a decimation ratio of 4:1. Figure 4-10 is a 64 point DWD of the original music data and Figure 4-11 is the corresponding band-select 64 point DWD output for the observation band, 5500 to 6500Hz. Spectral resolution over the observation band is not obtained for a 64 point DWD, however the "pseudo"-zoom process resolves the individual tones. For comparative purposes a 256 point DWD of the music data and a 256 point DWD, filtered over the observation band, of the music data are given, respectfully, in Figures 4-12 and 4-13. Observe, in the 256 point DWD, the presence of tones that are not present in the band-select 64 point DWD. These

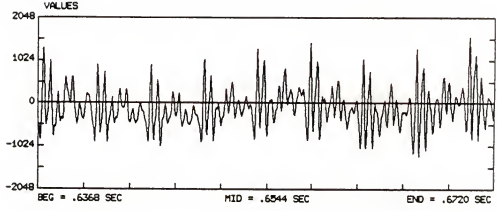

Figure 4-7. Music signal sampled at 20kHz.

```
BND PASS ELLIPTIC FILTER
PASS BAND RIPPLE                0.200 DB
PASS BAND EDGES              5500.001     6500.000 HZ
STOP BAND EDGES              5092.771     6853.496 HZ
SAMPLING FREQUENCY          20000.000 HZ
ATTENUATION                   -40.000 DB

     DENOMINATOR          NUMERATOR
1    1.000000E+00    1.235341E-02
2    2.325302E+00    2.466397E-02
3    5.470944E+00    4.917543E-02
4    6.855298E+00    5.984104E-02
5    8.228624E+00    7.337381E-02
6    6.093274E+00    5.984105E-02
7    4.323278E+00    4.917543E-02
8    1.629330E+00    2.466397E-02
9    6.228603E-01    1.235341E-02
```

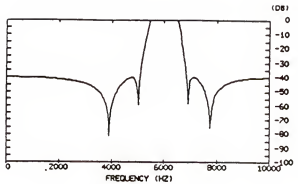

Figure 4-8. Bandpass filter for "psuedo"-zoom DWD example.

```
SAMPLING FREQUENCY          20000.000 HZ
LOW PASS CHEBYCHEV FILTER
STOP BAND ATTENUATION          40.000 DB
BAND EDGES                   1100.000

     DENOMINATOR          NUMERATOR
1    1.000000E+00    9.095963E-03
2   -6.315048E+00   -5.067343E-02
3    1.759886E+01    1.337898E-01
4   -2.824057E+01   -2.225362E-01
5    2.851816E+01    2.607683E-01
6   -1.854670E+01   -2.225362E-01
7    7.582170E+00    1.337897E-01
8   -1.780780E+00   -5.067343E-02
9    1.839127E-01    9.095962E-03
```

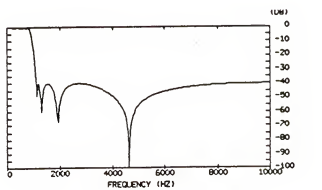

Figure 4-9. Lowpass filter for "psuedo"-zoom DWD example.

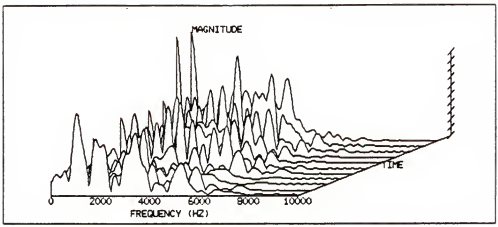

Figure 4-10. DWD of music data for N=64.

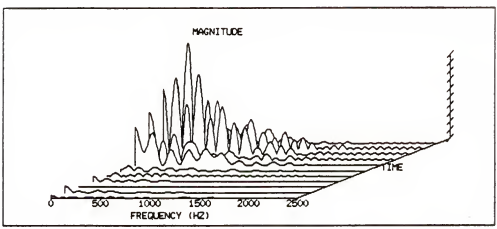

Figure 4-11. Band-select DWD of music data for N=64.

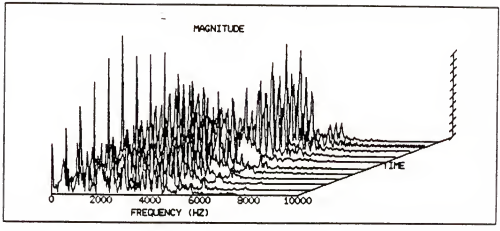

Figure 4-12. DWD of music data for N=256.

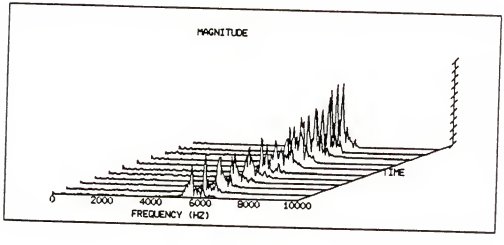

Figure 4-13. DWD of music data, filtered over the observation band, for N=256.

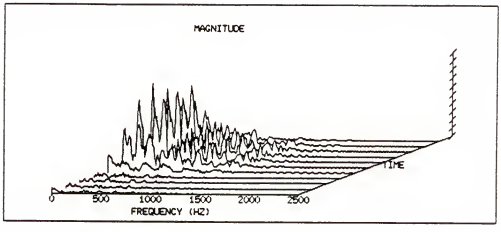

Figure 4-14. Band-select DWD of music data for N=256.

result from the cross-terms contributions between tones outside the observation band. Figure 4-14 is a 256 point band-select DWD where, as expected, the cross-term contributions have been elliminated.

To increase resolution and reduce computations the simplified band-select DWD of Figure 4-6 may be modified as follows:

partial decimation ⇒ improved attenuation of BPF;

lowpass filter (complex coefficients ⇒ SM-QRNS candidate);

decimation;

complex demodulation to "near" baseband;

kernel generation;

FFT.

Working with linear phase filters makes achieving a flat passband with sharp attenuation a difficult task; especially if the observation is very narrow relative to the total band. To prevent aliasing in the the DWD, real data sequences require that the signal be sampled at double the Nyquist rate and analytic signals require sampling to be at the Nyquist rate. For bandpass signals then aliasing can be prevented with a sampling frequency of four or eight times the bandwidth over which the spectrum is nonzero,

(depending on whether the signal is real or analytic). As such, for narrow observation bands away from baseband, partial decimation can be performed immediatly prior to bandpass filtering. By doing this high attenuation of the bandpass filter can be effected for a flat passband. For example, consider an observation band of 400Hz over a total band of 4KHz (as applicable to speech). For real data the sampling rate would exceed 16KHz. If a 4:1 decimation where performed after bandpass filtering, effecting a 4KHz sampling rate,the bandpass criteria to prevent aliasing would still be satisfied. After lowpass filtering then a 2:1 decimation may be performed producing an overall 8:1 decimation. Due to the increased attenuation of the bandpass filter, the required DWD "guard-band" is reduced; this translates to higher resolution in the "psuedo"-zoom DWD output.

If complex demodulation is used on the signal, the mixing process can be passed through an FIR lowpass filter as illustrated in Figure 4-15. Here the coefficients of the LPF are complex however decimation of the sequence can be performed prior to the mixing process, hence reducing the number of multiplies in the mixing process by the decimation ratio. Either the lowpass filter or the entire complex demodulation process may be implemented in the QRNS. In either case conversion to QRNS would be performed after bandpass filtering and partial decimation.

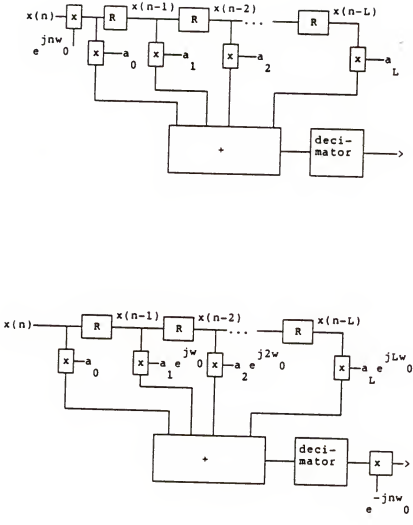

Figure 4-15. Multiply reduction a)Complex demodulation followed by lowpass filtering; b)Lowpass filtering with complex coefficients followed by complex demodulation.

4.4 Acousto-optic Wigner Processor

Many problems requiring enhanced time-frequency analysis require computational bandwidths well beyond those which can be obtained using conventional digital implementations. For example, electronic countermeasure (ECM) processors and a high bandwidth doppler radar, using a chirp envelope, fall into this category. To intercept these potential problems, an alternative technology must be considered. In this section an acousto-optic Wigner processor configuration is presented as a potential real-time signal processor.

Several papers have suggested producing a Wigner spectrum by optical means. Bartelt et.al. [32], in 1980 presented preliminary experimental results on an optical processor whose output is related to an approximation of the Wigner spectrum; it was called a "local spectrum." Their embodiment did not provide a means of producing the Wigner kernel. Kumar et al. performed a paper study of an optical Wigner processor [66]. They assumed the existence of an optical Wigner processor and conjectured its performance based on digital simulation. The simulator did not emulate the optical subsystem but simply signal in noise tests on the WD. This section will address more definative optical processor design issues. More specifically, in this section an acousto-optic specialized

Wigner distribution processor for operation at very high data rates will be developed.

Acousto-optic processing techniques have been successfully applied to spectrum analysis [67-69]. A laser beam provides a coherent source which is expanded and recollimated. The expanded beam passes through a surface acoustic wave device where the near surface interaction with a diffraction function produces the transmittance beam eminating from the SAW device.

To produce the diffraction pattern, a signal s(t) is applied to a transducer at one end of the SAW device. The transducer generates a surface acoustic wave with propagation delay $\tau=x/v$, where x is in the direction of propagation and v is the velocity of the acoustic wave [68-71]. The surface acoustic wave travelling along the cell effects a moving diffraction pattern of the form $s(t-x/v)$.

Given a laser source, the incident field assumes a Gaussian plane wave of the form

$$g(x)=a(x)e^{-j\omega_1 t}; \qquad (4\text{-}17)$$

$$a(x)=\sqrt{2/\pi}(1/a_0)e^{-x^2/a_0}$$

where ω_1 is the photon angular frequency c/λ and a_0 is the beam waist. The acousto-optic interaction produces a

refracted wave which when operating in the Bragg regime can be represented by a principle diffracted beam in quadrature with an undiffracted main beam [68-71]. That is, the transmittance field can be approximated as a phase modulation on the incident optical beam to produce a complex field amplitude of the following form

$$e(t,x)=a(x)e^{j\alpha s(t-x/v)} \qquad (4-18)$$

where α is an attenuation term. If the acoustic wave is small in magnitude, the diffracted beam can be represented by a first order Taylor series approximation on the exponent producing the transmittance field amplitude

$$e(t,x)=a(x)[1+j\alpha s(t-x/v)] \qquad (4-19)$$

which comprises the diffracted and undiffracted field components.

To achieve optimum diffraction efficiency, the magnitude of the photon momentum must be conserved. Photon mismatch of the incident beam and the transmittance wave is minimized when the optical wave is incident on the surface acoustic wave at the Bragg angle, $\theta_{Bn}=\sin^{-1}(\lambda/2n\Lambda)$, for Λ the acoustic wavelength and n the refractive index of the

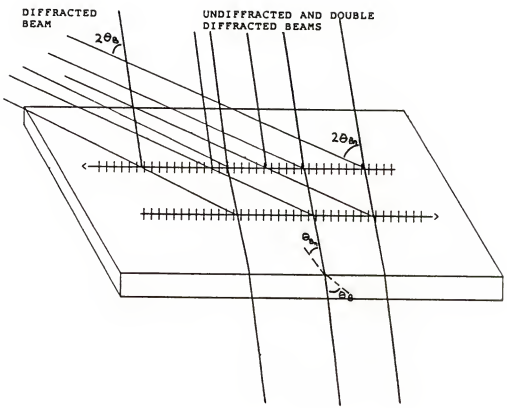

Figure 4-16. Bragg cell diffraction of counterpropagating waves.

cell material [68-71] (Figure 4-16). Given the acoustic wave propagation to be parallel to the cell edge, for an angle of incidence on the cell at the Bragg angle $\theta_B=\sin^{-1}(\lambda/2\Lambda)$, then according to Snell's law, the optical wave is incident on the acoustic wave in the material at the Bragg angle in the device material, given by θ_{Bn}. A portion of the light is diffracted from the incident beam at an angle $2\theta_{Bn}$. Again via Snell's law, the angle between the transmittance beams is $2\theta_B$.

The Wigner inner product is produced by imposing a double diffraction on the optical beam with two counterpropagating surface acoustic waves referenced from the cell center. Figure 4-17 illustrates the acousto-optic Wigner processor configuration. The light source is incident on Cl producing the diffracted wave complex amplitude

$$e(t,x)=a'(x)s(t-x/v) \qquad (4-20)$$

diffracted at an angle with respect to the undiffracted main beam of $2\theta_B$. The weighting term $a'(x)$ is a scaled, truncated form of $a(x)$ dependent primarily on the cell aperture, beam expansion characteristic and acoustic attenuation [68-71].

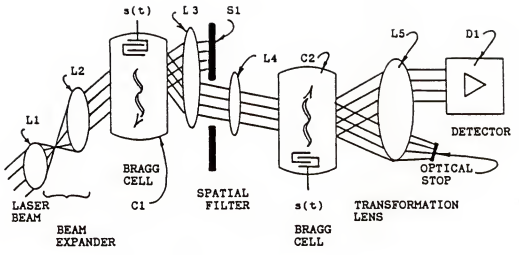

Figure 4-17. Optical Wigner processor configuration.

Spatial filtering to eliminate the undiffracted main beam is effected by L3, S1 and L4. Lens L3 images the transform of $e(t,x)$ onto S1. As $e(t,x)$ is diffracted at the angle $2\theta_B$ from the undiffracted beam, where θ_B is given above, they are spatially separated in the transform plane by an amount $f\lambda/\Lambda$. A slit at S1 passes only the diffracted beam. Lens L4 images the undiffracted beam onto cell C2 where a double diffraction is imposed by the counterpropagating acoustic wave at the Bragg angle. The transmittance field amplitude comprises a principle beam of the form

$$ee(t,x)=a'(x)s(t-x/v)s(t+x/v), \qquad (4-21)$$

diffracted from the main beam by $2\theta_B$. An optical stop at D1 is used to eliminate the undiffracted term. The complex amplitude at the back focal plane of the transform lens, L5, expresses as [refer to Appendix B for details]

$$W(t,x_f)=K\int_{-\infty}^{\infty}a'(x)s(t-x/v)s(t+x/v)e^{-jx(x_f/\lambda f)}dx. \qquad (4-22)$$

The spatial resolution in the transform plane is seen to be primarily dependent on the weighting term, $a'(x)$. If the distance transversed by the collimated beam is sufficiently

small relative to the length of the cell and the collimating lens aperture, the smoothing factor a'(x) approximates a constant over the length of the acoustic cell [72].

For analytic signals complex conjugation of the signal can be effected via quadrature modulation then SSB filtering [68]. For $s_R(t,x)$ and $s_I(t,x)$ the respective real and imaginary signal components, the quadrature modulated signal expresses as

$$s(t,x) = s_R(t,x)\cos\omega_c x - s_I(t,x)\sin\omega_c x \ . \qquad (4-23)$$

The transform of s(t,x), taken with respect to x, then expresses as

$$\{S_R(\omega-\omega_c)+S_R(\omega+\omega_c)\}-j\{S_I(\omega-\omega_c)-S_I(\omega+\omega_c)\}. \qquad (4-24)$$

Equation (4-24) can be seen as simply the transform of the demodulated signal around $-\omega_c$ and the transform of the complex conjugate around $+\omega_c$. Placing a screen at the focal point of the transform lens that passes only the term at $+\omega_c$ then inverse transforming the wave produces $s^*(t,x)e^{-j\omega_c x}$. The phase term isomorphically translates the output spectra an amount equivalent to ω_c in the Wigner plane which can be accomodated for at the detector.

Figure 4-18 indicates a hybrid optical guided-wave Wigner processor configuration. A planar waveguide is employed where surface acoustic waves are excited on a substrate by transducers placed at opposite ends of the device. Acoustic absorbers are placed between the transducer and the device edge to eliminate unwanted counterpropagating waves that result from reflections [68-71]. Geodesic lenses and Bragg modulators have successfully been fabricated in Ti-indiffused $LiNbO_3$ substrates [69]. GaAs substrates have received considerable attention toward the development of totally integrated optical processors; however, $LiNbO_3$ substrates maintain fewer loss characteristics than GaAs substrates [68,69]. The laser diode is butt-coupled to the waveguide edge and the input is coupled into the guide which forms the transmission medium. The major component problem is one of two-dimensional detection. To produce an output pattern as a function of t and x_f, a parallel readout linear detector array [68] can be coupled to the waveguide edge via fiber optics.

In the above optical guided-wave Wigner processor configuration, signals of small BT (B=bandwidth, T=time) values were assumed, as the proposed processor employs space-integrating techniques and is therefore limited by the delay line length, lens aperture, etc. For signals of

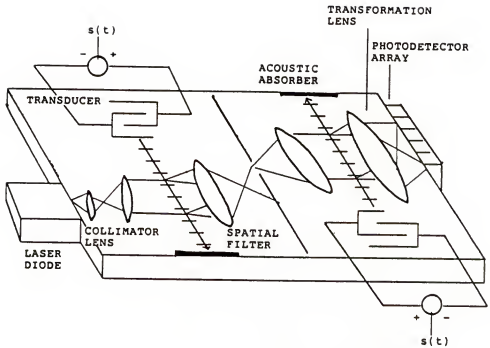

Figure 4-18. Hybrid optical guided-wave Wigner processor.

larger BT values, a more sophisticated processor employing time-integrating techniques is necessary; otherwise, the pseudo(or windowed)-WD must be observed, where the output is a frequency smoothed version of the WD. With the pWD, all time domain properties are maintained. Still, the pWD proves an attractive alternative to STFT techniques (Chapter 2).

CHAPTER 5
SUMMARY AND CONCLUSIONS

A time-dependent spectral estimator, called the Wigner distribution (WD), has been studied. Its relation to the corresponding one-dimensional time and frequency domains of a signal was addressed. The effects of filtering and modulation on the WD were investigated and shown to overcome some of the problems associated with the WD. A Gaussian weighting, for both the filter and window, were shown to achieve minimum variance in the weighted-WD plane. The relative constraint on the variance between window and filter required to maintain positivity was determined to be $\beta_w\beta_f<16$ for a normal distribution.

The double Fourier transform relationship between the WD and the AF was addressed. Accordingly the WD and AF both can be used for signal interpretation in much the same way the one-dimensional time and frequency domains of a signal are both used. For example, the duality between the AF and the WD can be applied to analysis of incomplete data from a nonstationary signal in a manner consistent with

that used between the one-dimensional time and frequency domains.

The DWD is an FFT based estimator and is therefore recognized as a viable competator to the STP. A detailed comparison between the DWD and the STP has been provided both analytically and experimentally. For comparative purposes, broadband communication signals were selected because of their conceptual accessibility and because their dispersion characterstics are highly time dependent.

The spDWD was interpreted in terms of filtering and modulation then used to improve DWD estimation of signal types for which the DWD produces poor results. Again, the signal examples were broadband communication signals. Application of the proposed window and filter shape were experimentally tested, then applied to a practical example. Variations of the filter variance for a window was shown to influence the estimation.

In practical applications the signals used in the examples would operate at higher frequencies than used in the simulations. The reason for selecting lower frequencies was primarily for illustrative clarity. That is, if the signals were carried at the higher frequencies, for 64 point and 256 point transforms a large number of consecutive time slices would be required to display one complete period of the signal. This would serve only to obscure the display without contributing to the analysis.

The programs used to effect the DWD and spDWD were written to operate interactively with the ILS subroutine SDI.

The DWD was shown to require sampling be at double the Nyquist rate. Therefore, to extract equivalent t-ω information, the displacement between window center positions in the DWD must be twice that of the STP. The same number of transforms are required for each, however the symmetry of the DWD kernel was shown to allow a reduction in each computation as will be discussed presently.

Several means of implementing the DWD and spDWD for various signal types were presented. For real input data the symmetry of both the DWD kernel and the corresponding DWD output kernel was shown to allow computation of two N point DWD time slices via one N/2 point FFT. This was accomplished by rearranging the kernel input to the FFT and applying the folding properties to the output. Using this result and modifying the approach slightly, computation of each DWD slice via an N/4 FFT seems to naturally follow. The spDWD kernel is also real symmetric and the same results apply to the spDWD.

For complex data processing, modular arithmetic was applied to the processor design to reduce computations. A numbering system called the quadratic residue numbering system (QRNS) was chosen because it allows an isomorphic mapping from the complex integer representation to the

representation of two-tuples where multiplication is completed via only two real multiplies and real one addition. This is in comparison to a standard complex multiply requiring four real multiplies and two real additions. The QRNS for the case of a single modulus was used in the design where the chosen modulus was of the form 2^n+1. With this choice of modulus the isomorphism to the two-tuple representation was a simple set of shift/add operations. Application of the single modulus QRNS was chosen such that the popular radix-2 and radix-4 FFT algorithms could be used in the processor implementation. That is, in a multi-moduli system, algorithms requiring a large number of nested multiplies require too much overhead associated with magnitude scaling. Naturally, even for the SM case some overhead is required but is minimal. A throughput comparison between the SM-QRNS and a conventional system showed the SM-QRNS, for a like amount of hardware, to have higher throughput even under worst case SM-QRNS conditions.

A band-select DWD for high frequency analysis was presented. The corresponding elimination of cross-term components was discussed. That is, cross-terms that result from frequencies outside the observation band did not appear in the band-select implementation. This result should be accounted for when interpreting the band-select output. To interpret the band-select output, comparison

with the spDWD (for the spDWD filter bandwidth less than the observation band) may be more appropriate. Recall, however, the spDWD effects a moving filter centered at each point (t,ω) whereas in the band-select implementation the filter remains centered about a fixed frequency.

Finally, an acousto-optic processor to effect the WD was presented. The processor implementation employed space-integrating techniques which contrains the input signal to those with small BT values; otherwise the pWD is observed. A processor using time-integrating techniques is needed for signals of larger BT values; however, a time-integrating processor implementation is not as staightforward, particularly for the case of the WD. A possible approach may be to effect the AF (which is much simpler for the case of time integration), then use the double Fourier transform relationship between the WD and the AF to produce the WD. A processor of this type would most likely employ both time and space-integrating techniques.

APPENDIX A
RESIDUE NUMBERING SYSTEM OVERVIEW

The definitions and relations presented here are fundamental to abstract algebra and can be found in a first year graduate text in the area. A particularly thorough treatment can be found in [73] and the notation used throughout this appendix has been made consistent with that used in [73] for textual cross-referencing.

The integer numbers form a monoid over both multiplication and addition where a monoid is defined as a triple (S,ϕ,u) in which S denotes a nonvacuous set, ϕ an associative binary composition in S, and u a unique unit element for which $a\phi u=a=u\phi a$ for all $a\varepsilon S$. With respect to addition, $\phi=+$ and $u=0$ while for multiplication, $\phi=\cdot$ and $u=1$; for both, $S=Z$. Any subset of the set Z that contains the unit element and is closed under the composition ϕ (i.e. for all $a,b\varepsilon S$, $a\phi b\varepsilon S$) is referred to as a submonoid (or monoid).

The residue numbering system (RNS) representation for integers is based on particular equivalence relations

termed congruences where an equivalence relation is any relation, E, on a set, S, which satisfies the following properties:

reflexive (aEa; aεS)
symmetry (aEb $\Rightarrow$ bEa; a,bεS)
transitivity (aEb and bEc $\Rightarrow$ aEc; a,b,cεS).

Now, given a monoid (S,ϕ,u), a congruence, $\equiv$, in S is an equivalence relation in S such that for any a,b,c,dεS, if a$\equiv$b and c$\equiv$d then aϕc$\equiv$bϕd. Two integers are defined as congruent modulo a third, denoted a$\equiv$b(modk), if a-b=nk where n is any integer.

The definition for congruence can be applied to the monoids (Z,+,0) and (Z,$\cdot$,1) where $\equiv$ is $\equiv$(modk). That is, for any integer, a, there exists integers b,k,a_k such that a=bk+a_k for k>0 and $0 \leq a_k < k$, which means a$\equiv a_k$(modk) where a_k defines a residue class (modulo k) given by $\bar{a}_k=\{a_k+nk;\ n\varepsilon Z\}$. The set of residue classes (modulo k) form a subgroup (or group) under ϕ (where ϕ can be used to denote multiplication or addition).

A group is a monoid in which all of its elements are invertible; that is, for every aεS there exists an inverse, call it a^{-1}, such that $aa^{-1}=u=a^{-1}a$ where $a^{-1}\varepsilon$S and is unique. A subgroup is a submonoid which acts as a group.

The group, called the quotient (or factor) group of Z relative to the $\equiv(\text{mod}k)$ and denoted Z/kZ, is of order k and, hence, is isomorphic to the group under ϕ of $Z_k=\{0,1,\ldots,k-1\}$.

Two monoids (groups), (S_1,ϕ_1,u_1) and (S_2,ϕ_2,u_2), are <u>isomorphic</u> if there exists a bijective (i.e. 1 to 1, onto) map $\alpha:S_1\rightarrow S_2$ such that for $a,b\varepsilon S_1$ $\alpha(a\phi_1 b)=\alpha(a)\phi_2\alpha(b)\varepsilon S_2$. The map α is called an isomorphism of S_1 onto S_2.

By definition then, $\alpha(u_1)=u_2$ for α an isomorphism. The isomorphism $\alpha:Z/kZ\rightarrow Z_k$ is given by $\alpha(\bar{a}_k)=a_k$ where each $a_k\varepsilon Z_k$ is termed a residue and equates to $a_k=a\text{mod}k$ for each $a\varepsilon Z$. The set of residues is loosely termed the additive (multiplicative) group of residue classes of Zmodulok and denoted Z_k. Although informal, this notation is common and will be used as given, unless stated otherwise, in the remaining discussion.

Given a finite set of integers $\{0,1,2,\ldots,M-1\} = Z_M$, each integer $a\varepsilon Z_M$ has a unique RNS representation given by:

$$a\rightarrow(a_1,a_2,\ldots,a_L);\ a_k=a\text{mod}p_k\varepsilon S_k \qquad \text{(A-1)}$$

where

$$M=\prod_{k=1}^{L} p_k^{e_k};\ \gcd(p_i,p_j)=1. \qquad \text{(A-2)}$$

Signed numbers, $a\varepsilon[-M/2,M/2)$, are represented as above if $a\geq 0$; if $a<0$, then $a_k=(M-|a|)\mathrm{mod}p_k$. Each residue set Z_k, $k=1,2,\ldots,L$ forms a subset of Z_M where each triple $(Z_k,+,0)$ defines a subgroup and $(Z_k,\cdot,1)$ a submonoid. Furthermore, each $(Z_k,+,0)$ is an abelian group, or a group for which + is commutative as well as associative in Z_k, and is referred to as the additive group of the ring $(Z_k,+,\cdot,0,1)$. Each triple $(Z_k,\cdot,1)$ is called the multiplicative monoid of the ring.

A ring $(S,\phi_1,\phi_2,u_1,u_2)$ is a structure comprising a nonvacuous, S, set together with two compositions ϕ_1, ϕ_2 in S and two distinguished unit elements $u_1,u_2\varepsilon S$ such that (S,ϕ_1,u_1) forms an abelian group (a group for which ϕ_1 is commutative as well as associative in S) and (S,ϕ_2,u_2) forms a monoid and for all $a,b,c\varepsilon S$, $a\phi_2(b\phi_1c)=a\phi_2b\phi_1a\phi_2c$ and $(b\phi_1c)\phi_2=b\phi_2a\phi_1c\phi_2a$. For example, the integers form a ring over addition and subtraction called the Gaussian ring of integers and denoted $(Z,+,\cdot,0,1)$. If the nonvacuous elements form an abelian subgroup of the monoid (S,ϕ_2,u_2), then $(S,\phi_1,\phi_2,u_1,u_2)$ defines a field. More precisely then, the residue sets $(Z_k,+,\cdot,0,1)$, $k=1,2,\ldots,L$ each form a field.

If S_1, $S_2,\ldots,S_L$ each form respective monoids (groups) under ϕ_1, $\phi_2,\ldots,\phi_L$, then their direct product $S_1XS_2X\ldots XS_L$ also forms a monoid (group) whose elements are the set of L-tuples, each denoted $(a_1, a_2,\ldots,a_L)$ for $a_i\varepsilon Z_i$

i=1,2,...,L under the following composition:

$$(a_1,a_2,\ldots,a_L)\phi(b_1,b_2,\ldots,b_L) = (a_1\phi_1 b_1, a_2\phi_2 b_2,\ldots,a_L\phi_L b_L). \qquad \text{(A-3)}$$

An important RNS property follows from this relation. That is, if $a,b,c\varepsilon Z_M$ and $c=a\phi b$ where $a\rightarrow(a_1,a_2,\ldots,a_L)$ and $b\rightarrow(b_1,b_2,\ldots,b_L)$ correspond to the L-tuple of residues each taken modulo p_k, k=1,2,...,L, then $c\rightarrow(c_1,c_2,\ldots,c_L)$ where $c_k=(a_k\phi b_k)\text{mod}p_k$, k=1,2,...,L.

In words, addition and multiplication can be computed via L concurrent modulo p_k operations without the need for interdigit communication. As a result, the RNS is referred to as a carry-free algebraic system where each operation $a_k\phi b_k$ is implemented as a table-lookup using high-speed semiconductor memory.

As a final point, in the RNS the dynamic range (given by A-2) may be exceeded during processing as long as the final output does not exceed the prespecified range. To illustrate, suppose the operation xy-z is to be performed on the following complex data: x=8+j3, y=3+j2, and z=10+j11. Direct multiplication of x and y yields the complex product (8+j3)(3+j2)=18+j25 from which 10+j11 is subtracted to produce a final value of 8+j14. If taken congruent modulo p where p=17, the product is

$\langle 18+j25 \rangle_{17} \equiv 1+j8$ which in a single modulus system (L=1) would produce ambiguous output as a stand-alone task. Coupled with the operation -zmodulop (z=10+j11), however, the output $\langle -9-j13 \rangle_{17} \equiv 8+j14$ is obtained, which is the expected output.

APPENDIX B
OPTICAL TRANSFORMATION PRINCIPLES

The transformation properties thin of lenses are well-known [74] and are given briefly below. Let $a_i(x,z)$ denote the input optical wave amplitude of a time harmonic wave travelling in the z direction (and independent of y) and incident on a thin lens. A standard assumption of a thin lens is that the transverse displacement of an optical beam passing through the lens be small such that a ray incident on the front surface of the lens at x exits from the back surface of the lens at x with a phase delay of the form

$$\phi(x)=kn\Delta(x)-k[\Delta(0)-\Delta(x)] \qquad (B-1)$$

where n is the refractive index of the lens material, $\Delta(x)$ denotes lens thickness, and $k=2\pi/\lambda$ where λ is the wavelength of the optical wave. In this case, the complex amplitude of the emergent optical wave is a phase delayed

version of the incident wave amplitude which satisfies the following relation

$$a_e(x)=e^{-j(k/2f)x^2}a_i(x)l(x) \qquad (B-2)$$

$$l(x)=\begin{cases} 1 \text{ inside the lens aperature} \\ 0 \text{ elsewhere} \end{cases}$$

for $k=2\pi/\lambda$, where λ is the free space wavelength of the incident field, n the refractive index of the lens material, and f the lens focal length as a function of n and lens curvature. (For a thin lens inclusion of the lens thickness imparts only a constant phase term of the form $e^{-jn\Delta(0)}$ where $\Delta(0)$ denotes the maximum thickness of the lens and is typically small.)

The Fresnel propagation approximation can be applied at the back focal plane of the lens. The Fresnel propagation approximation is essentially a first order binomial approximation on Huygen's principle, which states that each point on a wavefront may be reguarded as a new source of waves. Each wave element dx contributes to the complex amplitude of a wavefront at point P the equivalent of $(E_0a_e(x)/\lambda R)e^{jkR}dx$ producing a complex amplitude distribution which expresses as

$$a(x_p)=(E_0/\lambda)\int_{-\infty}^{\infty}(1/R)a_e(x)e^{jkR}dx \qquad \text{(B-3.a)}$$

where

$$R=[f^2+(x-x_p)^2]^{1/2}=f[1+((x-x_p)/\lambda)^2]^{1/2}$$
$$\approx z_p(1+(1/2)((x-x_p)/z_p)^2-(1/8)((x-x_p)/z_p)^4+\ldots) \qquad \text{(B-3.b)}$$

for E_0 a constant and R assumed >> λ. At the back focal plane of the transform lens, a first order approximation can be applied to R in the exponent where $z_p=f$. In the denominator small changes in R are not as important as in the exponent and R can be approximated by f. Equation (B-3) reduces to

$$a(x_f)=t(x_f)\int_{-\infty}^{\infty}a_e(x)e^{j[(k/2f)x^2-2\pi x(x_f/\lambda f)]}dx \qquad \text{(B-4)}$$

where

$$t(x_f)=(E_0/j\lambda f)e^{jkf}e^{jkx_f/2f}$$

for $x_f/\lambda f$ termed the spatial frequency coordinate. The term e^{jkf} is a constant phase term which can be neglected as it equates to unity when the square modulus is taken to compute light intensity. Substituting (B-2) in for $a_e(x)$

yields

$$a(x_f)=t(x_f)\int_{-\infty}^{\infty}a_i(x)l(x)e^{-j2\pi x(x_f/\lambda f)}dx. \qquad (B-5)$$

If $a_i(x)$ denotes a time harmonic transmittance wave amplitude incident on the lens imposed by a diffraction grating s(x) on a uniform monochromatic light source a distance d0 in front of the lens, the following relation holds[1]

$$A_i(x_f)=E_0S(x_f)e^{-j\pi\lambda d_0(x_f/\lambda f)^2} \qquad (B-6)$$

where $S(x_f)$ is the Fourier spectrum of s(x) and $A_i(x_f)$ is the Fourier spectrum of $a_i(x)$. Using this relation (B-5) becomes

$$a(x_f)=(E_0/\lambda f)t'(x_f)\int_{-\infty}^{\infty}s(x)l(x)e^{-j2\pi x(x_f/\lambda f)}dx \qquad (B-7)$$

where

$$t'(x_f)=e^{j(k\lambda^2/2)f(1-d0/f)(x_f/\lambda f)^2}.$$

For $d_0=f$, $t'(x_f)=1$ and (B-7) is the Fourier transform in $x_f/\lambda f$ of the diffraction function, s(x), subtended by the lens aperture.

APPENDIX C
GLOSSARY

AF (ambiguity function)

A two-dimensional representation of signal delay and doppler shift common to radar and sonar applications.

BT (bandwidth-time product)

The product of the effective frequency bandwidth, the frequency band over which the amplitude of the normalized spectrum exceeds $\exp[-\pi/4]$, and the effective time duration, the time duration over which the amplitude of the normalized signal exceeds $\exp[-\pi/4]$.

β (dispersion index)

An indicator of the spectral concentration of a signal. For FM signals, the dispersion index is defined as the product of the period of the modulating signal and its amplitude where the amplitude is the maximum frequency deviation from the carrier frequency of the FM signal.

CRNS (complex residue numbering system)

A finite ring of complex integers where each element of the ring is uniquely represented by a k-tuple of complex residues taken modulo a set of relative primes.

DFT (discrete Fourier transform)

The N samples taken at $\omega=2\pi k/N$ of the decompostion of an N point data sequence into linear combinations of $e^{i\omega n}$.

DWD (discrete Wigner distribution)

The discrete form of the Wigner distribution in both time and frequency and used in digital signal processing applications.

FFT (fast Fourier transform)

Any member of a class of algorithms used to efficiently compute the discrete Fourier transform by decomposing either the input or output sequence into integer powers.

QRNS (quadratic residue number system)

A complex residue representation where each prime, p, is chosen such that the quadratic $i^2 \equiv -1 \text{modulo} p$ has a solution which allows an isomorphism from each residue representaion to a two-tuple representation generated by the product set of integers modulo p with itself.

RNS (residue number system)

A finite ring of integers where each element of the ring is uniquely represented by a k-tuple of residues taken modulo a set of relative primes.

SM-QRNS (single-modulus quadratic residue number system)

The quadratic residue number system for the case of a single modulus (i.e. one prime).

spDWD (smoothed-pseudo discrete Wigner Distribution)

The discrete form of the weighted Wigner distribution in both time and frequency and used in digital signal processing applications.

STFT (short-time Fourier transform)

The Fourier transform of a windowed signal where stationarity is postulated over the length of the window as the window slides along the time axis.

STP (short-time periodogram)

The square magnitude of the short-time Fourier transform.

WD (Wigner distribution)

A two-dimensional representation of signal concentration over time and frequency which maintains the characteristic signal properties of the one-dimensional time and frequency domains yet attains negative values.

wWD (weighted Wigner distribution)

The resulting representation after moving averages, in the form of windowing and filtering operations, are applied to the Wigner distribution.

APPENDIX D
PROGRAM LISTING

```
C COS
C This program generates sinusoidal data, then makes it ILS
compatible
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256)
        REAL F0,TPI,RNU
        INTEGER M
        CHARACTER*8 NAMEI,NAMEO
        PRINT *, 'enter F0'
        READ *, F0
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the cos file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        PRINT *,'enter   data pnts per period (eg 1536.)'
        READ *,RNU
        TPI=8.*ATAN(1.)
        IREC=1
        M=-256
C set header block for ils compatibility
         READ(1)(ICOMMON(I), I=1,128)
         WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
250      READ(1,END=900)(ITRY(I), I=1,256)
C Generate cos data
         M=M+256
         DO 200 J=1,256
          ITRY(J)=1000.*COS(TPI*F0*(M+J)/RNU)
```

```
200       CONTINUE
          IREC=IREC+1
          WRITE(2,REC=IREC)(ITRY(I), I=1,256)
          GO TO 250
900       CONTINUE
          CLOSE(1)
          CLOSE(2)
          STOP
          END
```

```
C DWD
C This program takes ILS compatible sampled data, generates
the Wigner
C inner product, then outputs the IP as an ILS compatible
data file;
C in this form, when passed through the ILS program SDI the
output is
C the DWD of the original data.
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256)
        REAL FITRY(256)
        CHARACTER*8 NAMEI,NAMEO
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the DWD IP file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        IREC=1
C set header block for ils compatibility
         READ(1)(ICOMMON(I), I=1,128)
         WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
250      READ(1,END=900)(ITRY(I), I=1,256)
C generate DWD inner product
         DO 200 J=1,128
          FITRY(J)=FLOAT(ITRY(J))*FLOAT(ITRY(257-J))/1000.
          FITRY(257-J)=FITRY(J)
200      CONTINUE
         DO 300 J=1,256
          ITRY(J)=NINT(FITRY(J))
300      CONTINUE
         IREC=IREC+1
         WRITE(2,REC=IREC)(ITRY(I), I=1,256)
         GO TO 250
900      CONTINUE
         CLOSE(1)
         CLOSE(2)
         STOP
         END
```

```
C SQRMOD
C This program generates data from square pulse modulation
C on a carrier, then makes it ILS compatible
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256)
        REAL F0,TPI,RNU
        INTEGER M
        CHARACTER*8 NAMEI,NAMEO
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the square file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        PRINT *, 'enter the carrier frequency'
        READ *, F0
        PRINT *, 'enter   data points per period (eg
1536.)'
        READ *,RNU
        TPI=8.*ATAN(1.)
        IREC=1
        M=-256
C set header block for ils compatibility
         READ(1)(ICOMMON(I), I=1,128)
         WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
250      READ(1,END=900)(ITRY(I), I=1,256)
C Generate square pulse data
         M=M+256
         DO 200 J=1,256
          VAL=M+J
          HNU=RNU/2
983       IF(VAL .GT. RNU)THEN
           VAL=VAL-RNU
           GO TO 983
          ELSE
           IF(VAL .LE. HNU)THEN
            ITRY(J)=1000.*COS(TPI*F0*(M+J)/RNU)
           ELSE
            ITRY(J)=0.
           END IF
          END IF
200      CONTINUE
         IREC=IREC+1
```

```
      WRITE(2,REC=IREC)(ITRY(I), I=1,256)
      GO TO 250
900   CONTINUE
      CLOSE(1)
      CLOSE(2)
      STOP
      END
```

```
C DWD64
C This program takes ILS compatible sampled data, generates
the Wigner
C inner product, then outputs the IP as an ILS compatible
data file;
C in this form, when passed through the ILS program SDI the
output is
C the DWD of the original data.
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256)
        REAL FITRY(256)
        CHARACTER*8 NAMEI,NAMEO
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the inner product file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        IREC=1
C set header block for ils compatibility
         READ(1)(ICOMMON(I), I=1,128)
         WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
250      READ(1,END=900)(ITRY(I), I=1,256)
C generate DWD inner product
         DO 200 J=1,32
          FITRY(J)=FLOAT(ITRY(J))*FLOAT(ITRY(65-J))/1000.

FITRY(J+64)=FLOAT(ITRY(J+64))*FLOAT(ITRY(129-J))/1000.

FITRY(J+128)=FLOAT(ITRY(J+128))*FLOAT(ITRY(193-J))/1000.

FITRY(J+192)=FLOAT(ITRY(J+192))*FLOAT(ITRY(257-J))/1000.
          FITRY(65-J)=FITRY(J)
          FITRY(129-J)=FITRY(J+64)
          FITRY(193-J)=FITRY(J+128)
          FITRY(257-J)=FITRY(J+192)
200      CONTINUE
         DO 300 J=1,256
          ITRY(J)=NINT(FITRY(J))
300      CONTINUE
         IREC=IREC+1
         WRITE(2,REC=IREC)(ITRY(I), I=1,256)
         GO TO 250
```

```
900     CONTINUE
        CLOSE(1)
        CLOSE(2)
        STOP
        END
```

```
C DWDW
C This program takes ILS compatible sampled data,
C windows it, generates the Wigner
C inner product, then outputs the IP as an ILS compatible
data file;
C in this form, when passed through the ILS program SDI the
output is
C the DWD of the original data.
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256)
        REAL FITRY(256), WNDW(128),PITRY(256)
        CHARACTER*8 NAMEI,NAMEO
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the DWD IP file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        IREC=1
        READ(3,*)(WNDW(J), J=1,128)
C set header block for ils compatibility
         READ(1)(ICOMMON(I), I=1,128)
         WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
250      READ(1,END=900)(ITRY(I), I=1,256)
C generate DWD inner product
         DO 200 J=1,128
          PITRY(J)=FLOAT(ITRY(J))*WNDW(J)
          PITRY(257-J)=FLOAT(ITRY(257-J))*WNDW(J)
          FITRY(J)=PITRY(J)*PITRY(257-J)/1000.
          FITRY(257-J)=FITRY(J)
200      CONTINUE
         DO 300 J=1,256
          ITRY(J)=NINT(FITRY(J))
300      CONTINUE
         IREC=IREC+1
         WRITE(2,REC=IREC)(ITRY(I), I=1,256)
         GO TO 250
900      CONTINUE
         CLOSE(1)
         CLOSE(2)
         CLOSE(3)
         STOP
         END
```

```
C DWD64
C This program takes ILS compatible sampled data, generates
the Wigner
C inner product, then outputs the IP as an ILS compatible
data file;
C in this form, when passed through the ILS program SDI the
output is
C the DWD of the original data.
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256)
        REAL FITRY(256),PITRY(256),WNDW(32)
        CHARACTER*8 NAMEI,NAMEO
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the inner product file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        IREC=1
        READ(3,*)(WNDW(J), J=1,32)
C set header block for ils compatibility
         READ(1)(ICOMMON(I), I=1,128)
         WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
250      READ(1,END=900)(ITRY(I), I=1,256)
C generate DWD inner product
         DO 200 J=1,32
          PITRY(J)=FLOAT(ITRY(J))*WNDW(J)
          PITRY(J+64)=FLOAT(ITRY(J+64))*WNDW(J)
          PITRY(J+128)=FLOAT(ITRY(J+128))*WNDW(J)
          PITRY(J+192)=FLOAT(ITRY(J+192))*WNDW(J)
          PITRY(65-J)=FLOAT(ITRY(65-J))*WNDW(J)
          PITRY(129-J)=FLOAT(ITRY(129-J))*WNDW(J)
          PITRY(193-J)=FLOAT(ITRY(193-J))*WNDW(J)
          PITRY(257-J)=FLOAT(ITRY(257-J))*WNDW(J)
          FITRY(J)=PITRY(J)*PITRY(65-J)/1000.
          FITRY(J+64)=PITRY(J+64)*PITRY(129-J)/1000.
          FITRY(J+128)=PITRY(J+128)*PITRY(193-J)/1000.
          FITRY(J+192)=PITRY(J+192)*PITRY(257-J)/1000.
          FITRY(65-J)=FITRY(J)
          FITRY(129-J)=FITRY(J+64)
          FITRY(193-J)=FITRY(J+128)
          FITRY(257-J)=FITRY(J+192)
200      CONTINUE
```

```
      DO 300 J=1,256
       ITRY(J)=NINT(FITRY(J))
300   CONTINUE
      IREC=IREC+1
      WRITE(2,REC=IREC)(ITRY(I), I=1,256)
      GO TO 250
900   CONTINUE
      CLOSE(1)
      CLOSE(2)
      STOP
      END
```

```
C SCIRMOD2
C This program generates data from sin pulse modulation
C on a carrier, then makes it ILS compatible
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256)
        REAL F0,TPI,RNU,FMOD
        INTEGER M
        CHARACTER*8 NAMEI,NAMEO
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the sine pulse file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        PRINT *, 'enter the carrier frequency'
        READ *, F0
        PRINT *, 'enter   data points per period (eg
1536.)'
        READ *,RNU
        TPI=8.*ATAN(1.)
        PI=TPI/2
        IREC=1
        M=-256
C set header block for ils compatibility
         READ(1)(ICOMMON(I), I=1,128)
         WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
250      READ(1,END=900)(ITRY(I), I=1,256)
C Generate sine pulse data
         M=M+256
         DO 200 J=1,256
          VAL=M+J
          HNU=RNU/2
983       IF(VAL .GT. RNU)THEN
           VAL=VAL-RNU
           GO TO 983
          ELSE
            FMOD=1000.*COS(PI*VAL/RNU)
            ITRY(J)=FMOD*COS(TPI*F0*(M+J)/RNU)
          END IF
200      CONTINUE
         IREC=IREC+1
         WRITE(2,REC=IREC)(ITRY(I), I=1,256)
         GO TO 250
```

```
900     CONTINUE
        CLOSE(1)
        CLOSE(2)
        STOP
        END
```

```
C FMCOS
C This program generates data from FM sin pulse modulation
C on a carrier, then makes it ILS compatible
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256)
        REAL F0,TPI,RNU,FMOD
        INTEGER M
        CHARACTER*8 NAMEI,NAMEO
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the FM file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        PRINT *, 'enter the carrier frequency'
        READ *, F0
        PRINT *, 'enter   data points per period (eg
1536.)'
        READ *,RNU
        PRINT *, 'enter GAIN value'
        READ *, GAIN
        TPI=8.*ATAN(1.)
        PI=TPI/2
        IREC=1
        M=-256
C set header block for ils compatibility
         READ(1)(ICOMMON(I), I=1,128)
         WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
250      READ(1,END=900)(ITRY(I), I=1,256)
C Generate FM data
         M=M+256
         DO 200 J=1,256
          VAL=M+J
983       IF(VAL .GT. RNU)THEN
           VAL=VAL-RNU
           GO TO 983
          ELSE
            FMOD=1.+COS(PI*VAL/RNU)
            ITRY(J)=GAIN*COS(TPI*F0*FMOD*VAL/RNU)
          END IF
200      CONTINUE
         IREC=IREC+1
         WRITE(2,REC=IREC)(ITRY(I), I=1,256)
```

```
      GO TO 250
900   CONTINUE
      CLOSE(1)
      CLOSE(2)
      STOP
      END
```

```
C CHRP
C This program generates chirp data, then makes it ILS
compatible
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256)
        REAL F0,TPI,RNU
        INTEGER M
        CHARACTER*8 NAMEI,NAMEO
        PRINT *, 'enter F0'
        READ *, F0
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the chirp file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        PRINT *, 'enter   data points per period (eg
1536.)'
        READ *,RNU
        TPI=8.*ATAN(1.)
        IREC=1
        M=-256
C set header block for ils compatibility
         READ(1)(ICOMMON(I), I=1,128)
         WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
250      READ(1,END=900)(ITRY(I), I=1,256)
C Generate chirp data
         M=M+256
         DO 200 J=1,256
          ITRY(J)=1000.*COS(TPI*F0*((M+J)/RNU)**2/2.)
200      CONTINUE
         IREC=IREC+1
         WRITE(2,REC=IREC)(ITRY(I), I=1,256)
         GO TO 250
900      CONTINUE
         CLOSE(1)
         CLOSE(2)
         STOP
         END
```

```
C FMPULS
C This program generates data from square pulse FM
modulation
C on a carrier, then makes it ILS compatible
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256)
        REAL F0,TPI,RNU,FMOD
        INTEGER M
        CHARACTER*8 NAMEI,NAMEO
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the FM file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        PRINT *, 'enter the carrier frequency'
        READ *, F0
        PRINT *, 'enter   data points per period (eg
1536.)'
        READ *,RNU
        PRINT *, 'enter GAIN'
        READ *, GAIN
        TPI=8.*ATAN(1.)
        PI=TPI/2
        IREC=1
        M=-256
C set header block for ils compatibility
         READ(1)(ICOMMON(I), I=1,128)
         WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
250      READ(1,END=900)(ITRY(I), I=1,256)
C Generate pulse FM data
         M=M+256
         DO 200 J=1,256
          VAL=M+J
          HNU=RNU/2
983       IF(VAL .GT. RNU)THEN
           VAL=VAL-RNU
           GO TO 983
          ELSE
           IF(VAL .LE. HNU)THEN
            FMOD=1000.
           ELSE
            FMOD=500.
```

```
          END IF
         END IF
           ITRY(J)=GAIN*COS(TPI*F0*FMOD*VAL/RNU)
200      CONTINUE
         IREC=IREC+1
         WRITE(2,REC=IREC)(ITRY(I), I=1,256)
         GO TO 250
900      CONTINUE
         CLOSE(1)
         CLOSE(2)
         STOP
         END
```

```
C cos2
C This program generates data from the sum of two
sinusoids,
C then makes it ILS compatible
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256)
        REAL F0,TPI,RNU,F1
        INTEGER M
        CHARACTER*8 NAMEI,NAMEO
        PRINT *, 'enter F0'
        READ *, F0
        PRINT *, 'enter F1'
        READ *, F1
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the dual cosine file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        PRINT *, 'enter   data points per period (eg
1536.)'
        READ *,RNU
        TPI=8.*ATAN(1.)
        IREC=1
        M=-256
C set header block for ils compatibility
        READ(1)(ICOMMON(I), I=1,128)
        WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
250     READ(1,END=900)(ITRY(I), I=1,256)
C Generate cos2 data
        M=M+256
        DO 200 J=1,256

ITRY(J)=1000.*(COS(TPI*F0*(M+J)/RNU)+COS(TPI*F1*(M+J)/RNU))
200     CONTINUE
        IREC=IREC+1
        WRITE(2,REC=IREC)(ITRY(I), I=1,256)
        GO TO 250
900     CONTINUE
        CLOSE(1)
        CLOSE(2)
        STOP
        END
```

```
C SPDWD
C This program takes ILS compatible sampled data, generates
the SPDWD
C inner product, then outputs the IP as an ILS compatible
data file;
C in this form, when passed through the ILS program SDI the
output is
C the SPDWD of the original data.
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256),IOUT(256),SMV
        REAL TTRY(768),RIP
        CHARACTER*8 NAMEI,NAMEO
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the SPDWD IP file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        IREC=1
        WRITE(6,11)
11      FORMAT(1X, 'enter the smoothing parameter value')
        READ(5,*)SMV
        RSMV=REAL(SMV)
        WRITE(6,12)
12      FORMAT(1X, 'enter the necessary magscale for ILS
compatibility')
        READ(5,*)DGAIN
C set header block for ils compatibility
         READ(1)(ICOMMON(I), I=1,128)
         WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
         BLOCK=0
250      READ(1,END=900)(ITRY(I), I=1,256)
         BLOCK=1+BLOCK
         IF(BLOCK .EQ. 1)THEN
          DO 141 J=1,256
           TTRY(256+J)=REAL(ITRY(J))
141       CONTINUE
          GO TO 250
         ELSE
          IF(BLOCK .EQ. 2)THEN
           DO 142 J=1,256
            TTRY(512+J)=REAL(ITRY(J))
142        CONTINUE
```

```
C compute SPDWD kernel for first block=0
           DO 143 J=257,512
            RIP=0.
            DO 144 M=257-J,SMV
             RIP=TTRY(M+J)*TTRY(M+768-J)/(RSMV*DGAIN)+RIP
144         CONTINUE
            IOUT(J-256)=NINT(RIP)
143        CONTINUE
           IREC=IREC+1
           WRITE(2,REC=IREC)(IOUT(I), I=1,256)
           GO TO 250
          ELSE
C generate DWD inner product
           DO 200 J=1,256
            TTRY(J)=TTRY(256+J)
            TTRY(256+J)=TTRY(512+J)
            TTRY(512+J)=REAL(ITRY(J))
200        CONTINUE
           DO 211 J=257,512
            RIP=0.
            DO 212 M=-SMV,SMV
             RIP=TTRY(M+J)*TTRY(M+768-J)/(RSMV*2000.)+RIP
212         CONTINUE
            IOUT(J-256)=NINT(RIP)
211        CONTINUE
           IREC=IREC+1
           WRITE(2,REC=IREC)(IOUT(I), I=1,256)
           GO TO 250
          END IF
         END IF
C compute for last block
900      DO 341 J=1,256
          TTRY(J)=TTRY(256+J)
          TTRY(256+J)=TTRY(512+J)
341      CONTINUE
         DO 342 J=257,512
          RIP=0.
          DO 343 M=-SMV,512-J
           RIP=TTRY(M+J)*TTRY(M+768-J)/(RSMV*DGAIN)+RIP
343       CONTINUE
          IOUT(J-256)=NINT(RIP)
342      CONTINUE
         IREC=IREC+1
         WRITE(2,REC=IREC)(ITRY(I), I=1,256)
         CLOSE(1)
         CLOSE(2)
         STOP
         END
```

```
C CHRP2
C This program generates dual chirp data, then makes it ILS
compatible
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256)
        REAL F0,F1,TPI,RNU
        INTEGER M
        CHARACTER*8 NAMEI,NAMEO
        PRINT *, 'enter F0'
        READ *, F0
        PRINT *, 'enter F1'
        READ *, F1
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the chirp file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        PRINT *,'enter   data pnts per period'
        READ *,RNU
        TPI=8.*ATAN(1.)
        IREC=1
        M=-256
C set header block for ils compatibility
        READ(1)(ICOMMON(I), I=1,128)
        WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
250     READ(1,END=900)(ITRY(I), I=1,256)
C Generate chirp data
        M=M+256
        DO 200 J=1,256
         ITRY(J)=1000.*COS(TPI*F0*((M+J)/RNU)**2/2.)
     1          +1000.*COS(TPI*F1*((M+J)/RNU)**2/2.)
200     CONTINUE
        IREC=IREC+1
        WRITE(2,REC=IREC)(ITRY(I), I=1,256)
        GO TO 250
900     CONTINUE
        CLOSE(1)
        CLOSE(2)
        STOP
        END
```

```
C ECHO2D
C This program generates data for two time displaced FM
signals,
C at different frequencies, then makes it ILS compatible
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256),ITRY2(256)
        REAL F0,TPI,RNU,GAIN,GAINE
        INTEGER M,DISP
        CHARACTER*8 NAMEI,NAMEO
        PRINT *, 'enter F0'
        READ *, F0
        PRINT *, 'enter F1'
        READ *, F1
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the chirp file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        PRINT *, 'enter   data points per period (eg
1536.)'
        READ *,RNU
        PRINT *, 'enter time displacement value'
        READ *, DISP
        PRINT *, 'enter the signal amplitude'
        READ *, GAIN
        PRINT *, 'enter the echo ampl'
        READ *, GAINE
        TPI=8.*ATAN(1.)
        IREC=1
        M=-256
C set header block for ils compatibility
        READ(1)(ICOMMON(I), I=1,128)
        WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
250     READ(1,END=900)(ITRY(I), I=1,256)
C Generate chirp data
        M=M+256
        M2=M-DISP
        DO 200 J=1,256
         ITRY(J)=GAIN*COS(TPI*F0*((M+J)/RNU)**2/2.)
         ITRY2(J)=GAINE*COS(TPI*F1*((M2+J)/RNU)**2/2.)
200     CONTINUE
        DO 300 J=1,256
```

```
          ITRY(J)=ITRY(J)+ITRY2(J)
300       CONTINUE
          IREC=IREC+1
          WRITE(2,REC=IREC)(ITRY(I), I=1,256)
          GO TO 250
900       CONTINUE
          CLOSE(1)
          CLOSE(2)
          STOP
          END
```

```
C ECHO
C This program generates data for two time displaced FM
signals,
C then makes it ILS compatible
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256),ITRY2(256)
        REAL F0,TPI,RNU
        INTEGER M,DISP
        CHARACTER*8 NAMEI,NAMEO
        PRINT *, 'enter F0'
        READ *, F0
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the chirp file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        PRINT *, 'enter   data points per period (eg
1536.)'
        READ *,RNU
        PRINT *, 'enter time displacement value'
        READ *, DISP
        TPI=8.*ATAN(1.)
        IREC=1
        M=-256
C set header block for ils compatibility
         READ(1)(ICOMMON(I), I=1,128)
         WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
250     READ(1,END=900)(ITRY(I), I=1,256)
C Generate chirp data
        M=M+256
        M2=M-DISP
        DO 200 J=1,256
         ITRY(J)=1000.*COS(TPI*F0*((M+J)/RNU)**2/2.)
         ITRY2(J)=1000.*COS(TPI*F0*((M2+J)/RNU)**2/2.)
200     CONTINUE
        DO 300 J=1,256
         ITRY(J)=ITRY(J)+ITRY2(J)
300     CONTINUE
        IREC=IREC+1
        WRITE(2,REC=IREC)(ITRY(I), I=1,256)
        GO TO 250
900     CONTINUE
```

```
CLOSE(1)
CLOSE(2)
STOP
END
```

```
C SPDWDW2
C This program takes ILS compatible sampled data,
C windows it, generates the SPDWD
C inner product, then outputs the IP as an ILS compatible
data file;
C in this form, when passed through the ILS program SDI the
output is
C the SPDWD of the original data.
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256),IOUT(256),SMV
        REAL TTRY(768),RIP,FLTR(256),WNDW(256),WNDWS(256)
        CHARACTER*8 NAMEI,NAMEO
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the SPDWD IP file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        IREC=1
        WRITE(6,11)
11      FORMAT(1X, 'enter the smoothing parameter value')
        READ(5,*)SMV
        RSMV=REAL(SMV)
        WRITE(6,12)
12      FORMAT(1X, 'enter the necessary magscale for ILS
compatibility')
        READ(5,*)DGAIN
C generate window and filter functions
        READ(3,*)(WNDW(J), J=1,257)
        DO 33 J=1,257
         WNDWS(J)=WNDW(J)*WNDW(257-J)
33      CONTINUE
        PRINT *, 'enter the Gaussian attenuation parameter'
        READ *, ALPHA
        DO 34 J=1,SMV
         R=-1.*ALPHA*J**2
         FLTR(J)=EXP(R)
34      CONTINUE
C set header block for ils compatibility
         READ(1)(ICOMMON(I), I=1,128)
         WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
         BLOCK=0
250      READ(1,END=900)(ITRY(I), I=1,256)
```

```
          BLOCK=1+BLOCK
          IF(BLOCK .EQ. 1)THEN
           DO 141 J=1,256
            TTRY(256+J)=REAL(ITRY(J))
141        CONTINUE
           GO TO 250
          ELSE
           IF(BLOCK .EQ. 2)THEN
            DO 142 J=1,256
             TTRY(512+J)=REAL(ITRY(J))
142         CONTINUE
C compute SPDWD kernel for first block=0
            DO 143 J=257,512
             RIP=0.
             DO 144 M=257-J,SMV
              IF(M .LT. 0)THEN
               MP=-1*M
              ELSE
               MP=M
              END IF
              IF(MP .EQ. 0)THEN
               RIP=RIP
              ELSE

RIP=TTRY(M+J)*TTRY(M+768-J)*FLTR(MP)/(RSMV*DGAIN)+RIP
              END IF
144          CONTINUE
             OUT=RIP*WNDWS(J-256)
             IOUT(J-256)=NINT(OUT)
143         CONTINUE
            IREC=IREC+1
            WRITE(2,REC=IREC)(IOUT(I), I=1,256)
            GO TO 250
           ELSE
C generate DWD inner product
            DO 200 J=1,256
             TTRY(J)=TTRY(256+J)
             TTRY(256+J)=TTRY(512+J)
             TTRY(512+J)=REAL(ITRY(J))
200         CONTINUE
            DO 211 J=257,512
             RIP=0.
             DO 212 M=-SMV,SMV
              IF(M .LT. 0)THEN
               MP=-1*M
              ELSE
               MP=M
              END IF
              IF(MP .EQ. 0)THEN
               RIP=RIP
              ELSE
```

```
RIP=TTRY(M+J)*TTRY(M+768-J)*FLTR(MP)/(RSMV*2000.)+RIP
              END IF
212           CONTINUE
              OUT=RIP*WNDWS(J-256)
              IOUT(J-256)=NINT(OUT)
211           CONTINUE
              IREC=IREC+1
              WRITE(2,REC=IREC)(IOUT(I), I=1,256)
              GO TO 250
             END IF
            END IF
C compute for last block
900         DO 341 J=1,256
             TTRY(J)=TTRY(256+J)
             TTRY(256+J)=TTRY(512+J)
341         CONTINUE
            DO 342 J=257,512
             RIP=0.
             DO 343 M=-SMV,512-J
                IF(M .LT. 0)THEN
                 MP=-1*M
                ELSE
                 MP=M
                END IF
                IF(MP .EQ. 0)THEN
                 RIP=RIP
                ELSE

RIP=TTRY(M+J)*TTRY(M+768-J)*FLTR(MP)/(RSMV*DGAIN)+RIP
              END IF
343          CONTINUE
             OUT=RIP*WNDWS(J-256)
             IOUT(J-256)=NINT(OUT)
342         CONTINUE
            IREC=IREC+1
            WRITE(2,REC=IREC)(ITRY(I), I=1,256)
            CLOSE(1)
            CLOSE(2)
            STOP
            END
```

```
C SPLUSN
C This program adds noise to data, then makes it ILS
compatible
        INTEGER*4 ICOMMON(128),M,NRIP
        INTEGER*2 ITRY(256)
        REAL F0,TPI,RNU,RIP,THR
        CHARACTER*8 NAMEI,NAMEO
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the signal plus noise file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        PRINT *, 'enter seed'
        READ *,M
        PRINT *, 'enter noise threshold'
        READ *,THR
        IREC=1
C set header block for ils compatibility
         READ(1)(ICOMMON(I), I=1,128)
         WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
250      READ(1,END=900)(ITRY(I), I=1,256)
C Generate noise data
         DO 200 J=1,256
          RIP=RAN(M)
          RIP=THR*RIP
          NRIP=NINT(RIP)
          ITRY(J)=ITRY(J)+NRIP
200      CONTINUE
         IREC=IREC+1
         WRITE(2,REC=IREC)(ITRY(I), I=1,256)
         GO TO 250
900      CONTINUE
         CLOSE(1)
         CLOSE(2)
         STOP
         END
```

```
C SPNGAU
C This program adds Gaussian noise to ILS data
        INTEGER*4 ICOMMON(128),M,NRIP
        INTEGER*2 ITRY(256),IGAU(256)
        REAL F0,TPI,RNU,RIP,STD,RMEAN
        CHARACTER*8 NAMEI,NAMEO,NAMEN
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the signal plus noise file name')
        READ(5,130)NAMEO
130     FORMAT(A8)
        WRITE(6,140)
140     FORMAT(1X, 'enter the noise file name')
        READ(5,150)NAMEN
150     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)

OPEN(UNIT=3,NAME=NAMEN,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        PRINT *, 'enter seed'
        READ *,M
        PRINT *, 'enter seed coefficient'
        READ *, GAIN
        PRINT *, 'enter the large number'
        READ *, LNU
        HLNU=LNU/2.
        PRINT *, 'enter the standard deviation'
        READ *, STD
        PRINT *, 'enter the mean'
        READ *, RMEAN
        IREC=1
C set header block for ils compatibility
        READ(1)(ICOMMON(I), I=1,128)
        WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
        WRITE(3,REC=IREC)(ICOMMON(I), I=1,128)
250     READ(1,END=900)(ITRY(I), I=1,256)
C Generate Gaussian noise via law of large numbers
        DO 199 I=1,256
         GA=0.
         DO 200 J=1,LNU
          RIP=RAN(M)
          RM=RIP*GAIN
          M=NINT(RM)
```

```
          GA=RIP+GA
200       CONTINUE
          RIP2=(GA-HLNU)*STD+RMEAN
          IGAU(I)=NINT(RIP2)
199       CONTINUE
          DO 210 J=1,256
           ITRY(J)=ITRY(J)+IGAU(J)
210       CONTINUE
          IREC=IREC+1
          WRITE(2,REC=IREC)(ITRY(I), I=1,256)
          WRITE(3,REC=IREC)(IGAU(I), I=1,256)
          GO TO 250
900       CONTINUE
          CLOSE(1)
          CLOSE(2)
          CLOSE(3)
          STOP
          END
```

```
C        RFILT
C       This program takes sampled data and filters it via
C       a recursive filter
C
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256)
        DIMENSION ZLOW(9),ZBAND(9),AL(9),PL(9),AB(9),PB(9)
        DIMENSION CC(10)
        CHARACTER*8 NAMEI,NAMEO
        CHARACTER*8 NAMEA,NAMEP
        WRITE(6,100)
100     FORMAT(1X, 'Enter the sampled data file to be
processed')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'Enter the processed data file name')
        READ(5,130)NAMEO
130     FORMAT(A8)
        WRITE(6,140)
140     FORMAT(1X, 'Enter the denom coeffs')
        READ(5,150)NAMEA
150     FORMAT(A8)
        WRITE(6,160)
160     FORMAT(1X, 'Enter the num coeffs')
        READ(5,170)NAMEP
170     FORMAT(A8)
        OPEN
(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)
        OPEN
(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
        OPEN (UNIT=3,NAME=NAMEA,STATUS='OLD',READONLY)
        OPEN (UNIT=4,NAME=NAMEP,STATUS='OLD',READONLY)
C
C       initialize params and read filter coeffs
C
        ICOSI=0
        IREC=1
        READ(3,*) (AB(I), I=1,9)
        READ(4,*) (PB(I), I=1,9)
C
C       set header block for new data file
C
        READ(1) (ICOMMON(I), I=1,128)
        WRITE(2,REC=IREC) (ICOMMON(I), I=1,128)
C
550     READ(1,END=900) (ITRY(I), I=1,256)
C
C       --------------------
C       Filter loop
```

```
C
      DO L=1,256
C
      TIM=0.
      DO IBP=2,9
      ZBAND(11-IBP)=ZBAND(10-IBP)
      TIM=TIM-AB(11-IBP)*ZBAND(11-IBP)
      END DO
      ZBAND(1)=TIM+FLOAT(ITRY(L))
      YI=0.
      DO IBPF=1,9
      YI=YI+ZBAND(IBPF)*PB(IBPF)
      END DO
C
C     gain
C
      ITRY(L)=NINT(YI)
      END DO
      IREC=IREC+1
      WRITE(2,REC=IREC) (ITRY(I), I=1,256)
      GOTO 550
900   CONTINUE
      CLOSE(1)
      CLOSE(2)
      CLOSE(3)
      CLOSE(4)
      STOP
      END
```

```
C DEMZOOM
C this program modulates ILS compatible data
        INTEGER*4 ICOMMON(128)
        INTEGER*2 ITRY(256)
        REAL FC,TPI,RNU,RN
        INTEGER M
        CHARACTER*8 NAMEI,NAMEO
        PRINT *, 'enter FC'
        READ *, FC
        PRINT *, 'enter period'
        READ *, RNU
        WRITE(6,100)
100     FORMAT(1X, 'enter the sampled data file')
        READ(5,110)NAMEI
110     FORMAT(A8)
        WRITE(6,120)
120     FORMAT(1X, 'enter the processed file name')
        READ(5,130)NAMEO
130     FORMAT(A8)

OPEN(UNIT=1,NAME=NAMEI,STATUS='OLD',FORM='UNFORMATTED',
     1  READONLY)

OPEN(UNIT=2,NAME=NAMEO,STATUS='NEW',FORM='UNFORMATTED',
     1  ACCESS='DIRECT',RECL=128)
C initialize parameters
        TPI=8.*ATAN(1.)
        IREC=1
        M=-256
C set header block for ils compatibility
         READ(1)(ICOMMON(I), I=1,128)
         WRITE(2,REC=IREC)(ICOMMON(I), I=1,128)
250      READ(1,END=900)(ITRY(I), I=1,256)
C heterodyning process
         M=M+256
         DO 200 J=1,256
          ITRY(J)=ITRY(J)*COS(TPI*FC*(M+J)/RNU)
200      CONTINUE
         IREC=IREC+1
         WRITE(2,REC=IREC)(ITRY(I), I=1,256)
         GO TO 250
900      CONTINUE
         CLOSE(1)
         CLOSE(2)
         STOP
         END
```

REFERENCES

[1] C.H. Page,"Instantaneous Power Spectra," Journal of Applied Physics, vol. 23, no. 1, pp. 103-106, Jan 1952.

[2] M.J. Levin, "Instantaneous Spectra and Ambiguity Functions," IEEE Trans. on Information Theory, vol. IT-10, no. 1, pp. 95-97, Jan 1964.

[3] A.W. Rihaczek, "Signal Energy Distribution in Time and Frequency," IEEE Trans. on Information Theory, vol. 14, no. 3, pp. 369-374, May 1968.

[4] L. Cohen, "Generalized Phase-Space Distribution Functions," J. Math. Phys., vol. 7, pp. 781-786, 1966.

[5] N.G. DeBruijn, "Uncertainty Principles in Fourier Analysis," Inequalities, Academic Press, New York, pp. 57-71, 1967.

[6] N.G. De Bruijn, " A Theory of Generalized Functions, with Applications to Wigner Distribution and Weyl Correspondence," Nieuw Archief voor Wiskunde (3), vol. 21, pp. 205-280, 1973.

[7] T.A.C.M. Claasen and W.F.G. Mecklenbrauker, "The Wigner Distribution - A Tool for Time-Frequency Signal Analysis, Part III: Relations with Other Time-Frequency Signal Transformations," Philips J. Res., vol. 35, pp. 372-389, 1980.

[8] C.P. Janse and J.M. Kaizer, "Time-Frequency Distributions of Loudspeakers: the Application of the Wigner Distribution," J. Audio Engr. Soc., vol. 31, no. 4, pp. 198-223, 1983.

[9] P. Flandrin, "Some Features of Time-Frequency Representations of Multicomponent Signals," Proc. IEEE Int. Conf. on Acous., Speech, Signal Processing, vol. 41B, pp. 7.1-7.4, 1986.

[10] F. Peyrin and R. Prost, "A Unified Definition for the Discrete-Time, Discrete-Frequency, and Discrete-Time/Frequency Wigner Distributions," IEEE Trans. on Acous., Speech, Signal Processing, vol. 34, no. 4, pp. 858-867, August 1986.

[11] W. Martin and P. Flandrin, "Wigner-Ville Spectral Analysis of Nonstationary Processes," IEEE Trans. on Acous., Speech, Signal Processing, vol. 33, no. 6, pp. 1461-1470, Dec 1985.

[12] E. Wigner, "On the Quantum Correction For Thermodynamic Equilibrium," Physical Review, vol. 40, pp. 749-759, June 1932.

[13] J. Ville, "Theory and Applications of the Notion of a Complex Signal," Cables and Transmission, vol. 2, pp.61-74, 1948.

[14] T.A.C.M. Claasen and W.F.G. Mecklenbrauker, "The Wigner Distribution - A Tool for Time-Frequency Signal Analysis, Part I: Continuous-Time Signals," Philips J. Res., vol. 35, pp. 217-250, 1980.

[15] T.A.C.M. Claasen and W.F.G. Mecklenbrauker, "The Wigner Distribution - A Tool for Time-Frequency Signal Analysis, Part II: Discrete-Time Signals," Philips J. Res., vol. 35, pp. 276-300,1980.

[16] P. Flandrin and B. Escudie, "An Interpretation of the Wigner-Ville Distribution," Signal Processing, vol. 6, pp. 27-36, 1984.

[17] A.J.E.M. Janssen and T.A.C.M. Classen, "On the Positivity of Time-Frequency Distributions," IEEE Trans. on Acous., Speech, Signal Processing, vol. 33, no. 4, pp. 1029-1032, August 1985.

[18] T.A.C.M. Claasen and W.F.G. Mecklenbrauker, "The Aliasing Problem in Discrete-Time Wigner Distributions," IEEE Trans. on Acous., Speech, Signal Processing, vol. 31, no. 5, pp. 1067-1072, Oct. 1983.

[19] P. Flandrin and W. Martin, "Psuedo-Wigner Estimators for the Analysis of Non-stationary Processes," Proc. IEEE Acous., Speech, Signal Processing, Spectrum Est. Workshop II, vol. 2, pp. 181-185, 1983.

[20] D. Chester and J. Wilbur, " Time and Spatial Varying CAM and AI Signal Analysis Using the Wigner Distribution," Proc. IEEE Int. Conf. on Acous., Speech, Signal Processing, vol. 3, pp. 1045-1048, 1985.

[21] D. Chester, F.J. Taylor, and M. Doyle, "On the Wigner Distribution," Proc. IEEE Int. Conf. on Acous., Speech, Signal Processing, vol. 31, pp. 491-494, 1983.

[22] H. Van Maanen, "Duplication of the Sampling Frequency of Periodically Sampled Signals for the Calculation of the Discrete Wigner Distribution," J. Audio Eng. Soc., vol. 33, no. 11, Nov 1985.

[23] R.L. Hudson, "When is the Wigner Quasi-Probability Density Non-Negative?," Reports on Mathematical Physics, vol. 6, no. 2, pp. 449-450, 1974.

[24] D. Chester, F. Taylor, and M. Doyle, "Applications of the Wigner Distribution to Speech Processing," IEEE Workshop on Spectral Est., vol. 3, pp. 98-102, 1983.

[25] D. Preis, "Phase Distortion and Phase Equalization in Audio Signal Processing-A Tutorial Review," J. Audio Eng. Soc., vol. 30, no. 11, pp. 774-791, Nov 1982.

[26] W. Martin and P. Flandrin, "Detection of Changes of Signal Structure by Using the Wigner-Ville Spectrum," Signal Processing, vol. 8, no. 2, pp. 215-233, 1985.

[27] L. Jacobson and H. Wechsler, "The Composite Psuedo Wigner Distribution (CPWD): a Computable and Versatile Approximation to the Wigner Distribution (WD)," Proc. Int. Conf. on Acous., Speech, Signal Processing, vol. 1, pp. 254-256, 1983.

[28] B.V.K. Vijaya Kumar and C.W. Carrol, "Performance of Wigner Distribution Function Based Detection Methods," Optical Engineering, vol. 23, no. 6, pp. 732-737, Dec. 1984.

[29] M.J. Bastiaans, "The Wigner Distribution Function Applied to Optical Signals and Systems," Optics Communications, vol. 25, no. 1, pp. 26-38, April 1978.

[30] H.O. Bartelt, K.H. Brenner and A.W. Lohmann, "The Wigner Distribution Function and its Optical Production," Optics Communications, vol. 32, no. 1, pp. 32-38, Jan. 1980.

[31] B.E.A. Saleh and N.S. Subotic, "Time-Variant Filtering of Signals in the Mixed Time-Frequency Domain," IEEE Trans. on Acous., Speech, Signal Processing, vol. 33, no. 6, pp. 1479-1485, Dec 1985.

[32] G.F. Bourdreaux-Bartels and T.W. Parks, "Time-varying Filtering and Signal Estimation Using Wigner Distribution Filtering and Synthesis Techniques," IEEE Trans on Acous., Speech, Signal Processing, vol. 34, no. 3, pp. 442-451, June 1986.

[33] M.I Skolnik, Introduction to Radar Systems, McGraw-Hill, New York, 1980.

[34] D.S.K. Chan, "A Non-aliased Discrete-time Wigner Distribution for Time-Frequency Signal Analysis," Proc. IEEE Int. Conf. Acous., Speech, Signal Processing, pp. 1333-1336, May 1982.

[35] F.G. Stremler, Introduction to Communication Systems, Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1977.

[36] D.B. Chester, "The Wigner Distribution and Its Application to Speech Recognition and Analysis," Ph.D. Dissertation, University of Cincinnati, Cincinnati, Ohio, 1982.

[37] A.A. Giordano and F.M. Hsu, Least Square Esimation with Applications to Digital Signal Processing, John Wiley and Sons, Inc., New York, 1985.

[38] T.S. Durrani, R. Chapman, T. Willey, "Systolic Processor for Computing the Wigner Distribution," Elect. Lett., vol. 19, no. 3, pp. 790-791, June 1983.

[39] P. Flandrin, W. Martin and M. Zakharia, "On a Hardware Implementation of the Wigner-Ville Transform," Int. Conf. on Digital Signal Processsing, Florence, Italy, Sept. 1984.

[40] M.T. Heideman and C.S. Burrus, "A Bibliography of Fast Transform and Convolution Algorithms, II," Department of Electrical Engineering Technical Report No. 8402, Rice University, Houston, TX 77251-1892, Feb. 24, 1984.

[41] A.V. Oppenheim and R.W. Schafer, Digital Signal Processing, Prentice Hall, Inc., Englewood-Cliffs, New Jersey, 1975.

[42] R.K. Otnes and L. Enochson, Digital Time Series Analysis, John Wiley and Sons, Inc., New York, 1972.

[43] J.H. Mclellan and C.M. Rader, Number Theory in Digital Signal Processing, Prentice-Hall, Inc., 1979.

[44] R.E. Blahut, Fast Algorithms for Digital Signal Processing, Addison-Wesley, Inc., Reading, Massachusettes, 1984.

[45] S. Winograd, "On Computing the Discrete Fourier Transform," Math. Comp., vol. 32, pp. 175-199, Jan. 1978.

[46] S. Winograd, "On the Multiplicative Complexity of the Discrete Fourier Transform," Advances in Mathematics, vol. 32, no. 2, pp. 83-117, May 1979.

[47] S. Winograd, Arithmetic Complexity of Computation, SIAM CBMS-NSF Series, no. 32, Philidelphia, 1980.

[48] C.S. Burrus and T.W. Parks, DFT/FFT and Convolution Algorithms, John Wiley and Sons, Inc., New York, 1985.

[49] P. Duhamel and H. Hollman, "Split-Radix FFT Algorithm," Electronic Letters, vol. 20, no. 1, pp.14-16, Jan. 1984.

[50] P. Duhamel, "Implementation of 'Split-Radix' FFT Algorithms for Complex, Real, and Real Symmetric Data," IEEE Trans. on Acous., Speech, Signal Processing, vol. 34, pp. 285-295, April 1986.

[51] H.V. Sorensen, M.T. Heideman, C.S. Burrus, "On Calculating the Split-Radix FFT," IEEE Trans. on Acous., Speech, Signal Processing, vol. 34, pp. 152-156, Feb. 1986.

[52] F.J. Taylor, "Residue Arithmetic: A Tutorial with Examples," IEEE Computer Magazine, vol. 17, no. 5, pp. 50-62, May 1984.

[53] F.J. Taylor and C. Huang, "A Comparison of DFT Algorithms Using a Residue Architecture," Computer and Electrical Engineering (England), vol. 8, no. 3, pp. 161-171, Sept. 1981.

[54] F.J. Taylor, G. Papadourakis, A. Skavantzos, and A. Stouraitis, "A Radix-4 FFT Using Complex RNS Arithmetic," IEEE Trans. on Computers, vol. C-34, no. 6, pp. June 1985.

[55] M.C. Vanwormhoudt, "Structural Properties of Complex Residue Rings Applied to Number Theoretic Fourier Transforms," IEEE Trans. on Acous., Speech, Signal Processing, vol. 26, pp. 1140-1151, Feb. 1978.

[56] Shu-Hung Leung, "Application of Residue Number System to Complex Digital Filters," Proc. 15th Asilomar Conf. Circuits Systems and Computers, vol. 1, pp. 70-74, Nov. 1981.

[57] J.V. Krosmeier and W.K. Jenkins, "Error Detection and Correction in Quadratic Residue Number System," IEEE 26th Midwest Symposium on Circuits and Systems, Pueblo, Mexico, Aug. 1983.

[58] F.J. Taylor, "A Single Modulus Complex ALU for Signal Processing," IEEE Trans. on Acous., Speech, Signal Processing, vol. 33, no. 5, pp. 1302-1315, Oct. 1985.

[59] J. Wilbur and F.J. Taylor, "High-speed Wigner Processing Based on a Single Modulus Quadratic Residue Numbering System," Proc. IEEE Intl. Conf. on Acous., Speech, Signal Processing, vol. 2, pp. 1037-1040, April 1985.

[60] R-S Kao and F.J. Taylor, "Implementation of the Single-Modulus Complex ALU," IEEE Trans. on Cir. and Sys., vol. 11, pp. 450-456, 1987.

[61] F.J. Taylor, "On the Complex Arithmetic System (CRNS)," IEEE Trans. on Acous., Speech, Signal Processing, vol. 34, no. 6, Dec. 1986.

[62] H.L. Groginsky and G.A. Works, "A Pipeline Fast Fourier Transform," IEEE Trans. on Computers, vol. C-19, pp. 1015-1019, 1970.

[63] J.H. McClellan and R.J. Purdy, "Applications of Digital Signal Processing to Radar," in A.V. Oppenheim, ed., Applications of Digital Signal Processing, Prentice-Hall, Englewood Cliffs, Chapter 5, 1978.

[64] E.E. Swartzlander, Jr. and G. Hallnor, "High Speed FFT Processor Implementation," in M.E. Van Valkenburg, ed., VLSI Signal Processing, Pub. under the sponsership of IEEE Acous., Speech, Signal Processing Society, IEEE, Inc., New York, 1984.

[65] D. Chester and J. Wilbur, "Time and Spatial Varying CAM and AI Signal Analysis Using the Wigner Distribution," Proc. IEEE Int. Conf. on Acous., Speech, Signal Processing, vol. 3, pp. 1045-1048, 1985.

[66] B.V.K. Vijaya Kumar and C.W. Carrol, "Performance of Wigner Distribution Function Based Detection Methods," Optical Engineering, vol. 23, no. 6, pp. 793-800, Dec. 1984.

[67] C.S. Tsai, "A Review of Guided-Wave Acoustooptics with Applications to Real-Time Signal Processing," International Specialist Seminar on Case Studies in Advanced Signal Processing, Peebles, Scotland, Sep. 1979.

[68] N.J. Berg and J.N. Lee, ed., Acousto-optic Signal Processing, Marcel Dekker, Inc., New York, 1983.

[69] C.S. Tsai, "A Review of Guided-Wave Acoustooptics with Applications to Real-Time Signal Processing," International Specialist Seminar on Case Studies in Advanced Signal Processing, Peebles, Scotland, Sep. 1979.

[70] G. Ward, ed., Integrated Optics and Optical Communications, MSS Information Corp., New York, 1974.

[71] W.T. Welford, Optics, Oxford University Press, London, 1976.

[72] J. Wilbur and F.J. Taylor, "An Acoustooptic Wigner Processor For Time-varying Signal Analysis," IEEE Proceedings, vol. 75, no. 3, pp. 427-428, March 1987.

[73] N. Jacobson, Basic Algebra I
W.H. Freeman and Company, San Francisco, Cal., 1974.

[74] A. Papoulis, Systems and Transforms with Applications in Optics, McGraw-Hill Book Company, New York, 1968.

BIOGRAPHICAL SKETCH

JoEllen Wilbur was born in Fairfax County, Virginia, in 1959. She received the B.S. degree (magna cum laude) in electrical engineering from Virginia Polytechnic Institute and State University (Va Tech) in 1982. She then received the M.S. in electrical engineering from the University of Florida in 1984. She expects to receive the Ph.D. in August of 1987.

She received a CAD/CAM Graduate Institutional Fellowship in 1984, then an Army Research Office Graduate Fellowship in 1986. She is a member of Eta Kappa Nu, The Applied Optics Alumni Association (Va Tech), and IEEE.

JoEllen and her husband will both join the faculty of the Electrical and Computer Engineering Department at Clemson University in South Carolina in August of 1987.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Fred J. Taylor, Chairman
Professor of Electrical Engineering
and of Computer and Information Sciences

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Donald G. Childers
Professor of Electrical Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Antonio A. Arroyo
Associate Professor of Electrical
Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Jose C. Principe
Associate Professor of Electrical
Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

George Logothetis
Assistant Professor of Computer and Information Sciences

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate School and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

August 1987

Dean, College of Engineering

Dean, Graduate School